Software Networking and Interfaces

on Linux

Matt Turner

@mt165 | mt165.co.uk

SRECon, Singapore | June 2019


native wave

Outline

- Networking 101
- Interfaces
- Bridges
- Emergent Systems

This is not about

- IP, TCP, addressing
- SDN

Networking 101

Ethernet and ARP

Ethernet - L2 protocol

MAC Address - Media Access Control Address - Ethernet address, eg c0:ff:ee:be:ef:69

ARP - Address Resolution Protocol - DNS for ethernet: IP -> MAC

```
$ arp
Address                  HWtype  HWaddress           Flags Mask  Iface
192.168.0.239            ether   48:3b:38:01:6a:23   C           enp2s0
172.28.0.13              ether   02:42:ac:1c:00:0d   C           br-de368312f566
```

vLANs

Virtual LANs

IEEE 802.1q

Simulates multiple networks using one set of cables and switches

Each vLAN has a short numeric ID

Nested vLANs - IEEE 802.1ad, aka “q in q”

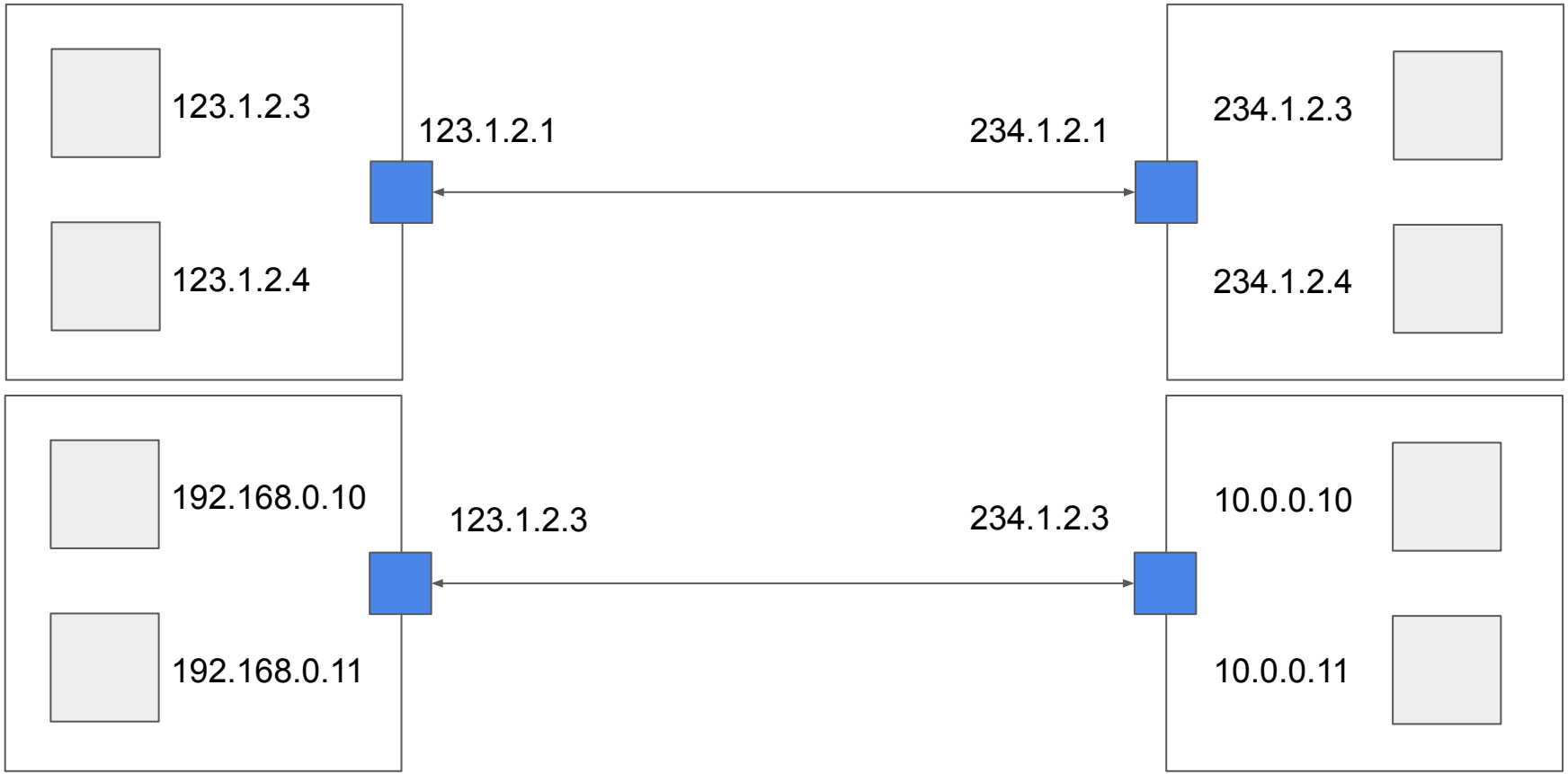
iptables

Linux kernel subsystem

Can do lots of things to packets as they pass through a system

Including: packet manipulation

NAT vs Routing



Route tables

Gives the next hop for a destination subnet.

```
$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         moon.lan        0.0.0.0         UG    3      0        0 enp2s0
loopback        localhost       255.0.0.0       UG    0      0        0 lo
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
```

Classful Routing

Class A	8 bit	0.0.0.0	127.0.0.1
Class B	16 bit	128.0.0.0	191.255.0.0
Class C	24 bit	192.0.0.0	223.255.255.0
Class D - multicast		224.0.0.0	239.255.255.255
Class E - reserved		240.0.0.0	255.255.255.255

Private Address Ranges

The “24-bit block”	8 bit prefix	10.0.0.0 – 10.255.255.255	1 class A
The “20-bit block”	12 bit prefix	172.16.0.0 – 172.31.255.255	16 class Bs
The “16-bit block “	16 bit prefix	192.168.0.0 – 192.168.255.255	256 class Cs
Loopback	8 bit prefix	127.0.0.0 - 127.255.255.255	1 class A

Classless Routing and CIDRs

Classless Inter-Domain Routing

Classful was too rigid and wasteful

Arbitrary ranges of addresses, notated as start address and size (as prefix mask)

CIDR notation: 192.168.42.0/24; 10.0.0.0/8

Broadcast, Multicast, Anycast

Address all, some, one host

- All hosts in a subnet
- Some hosts, which have opted in
- One arbitrary host from a set

DHCP

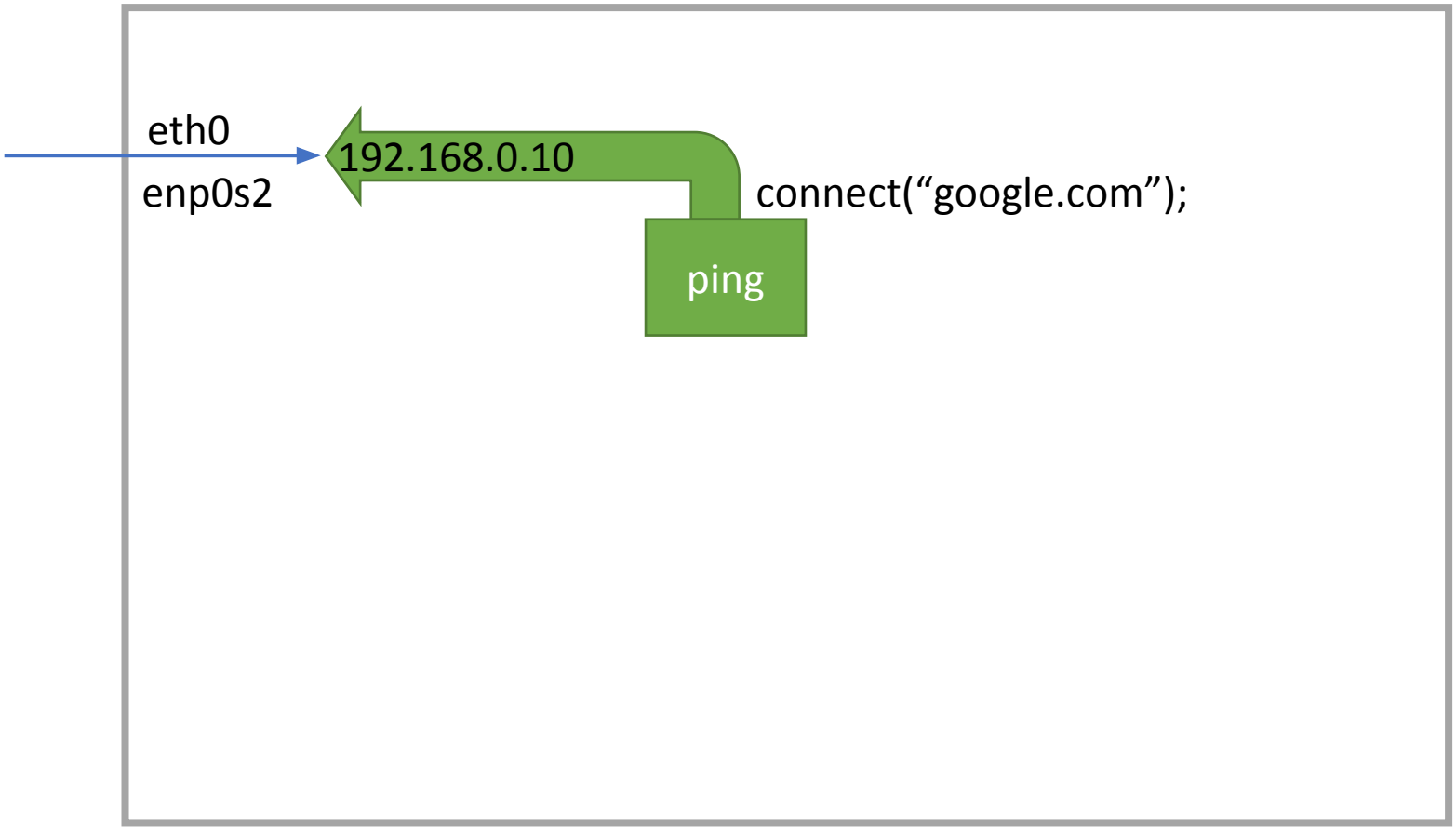
Asks a central server to allocate you an IP address

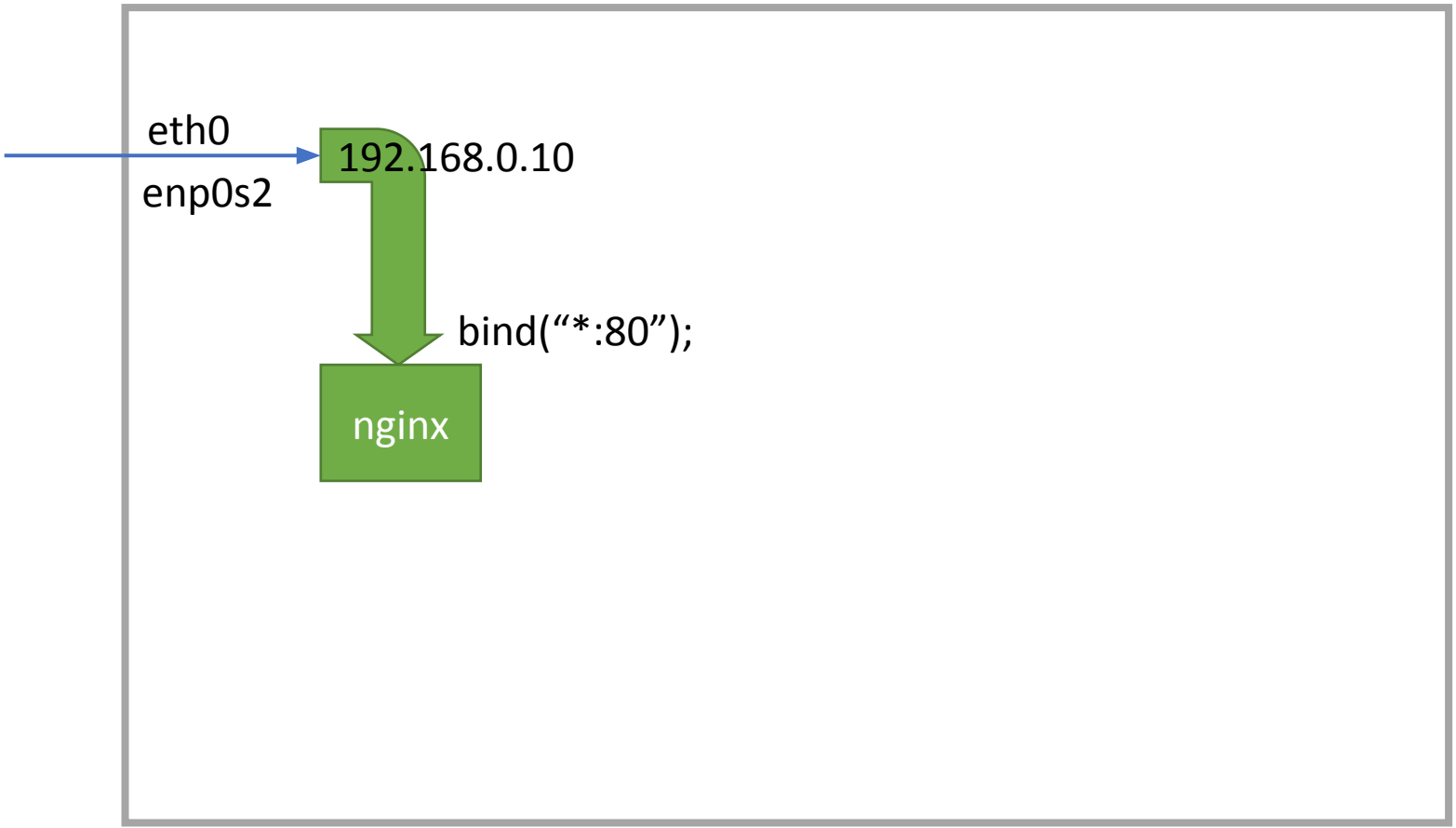
Based on an **ethernet** broadcast

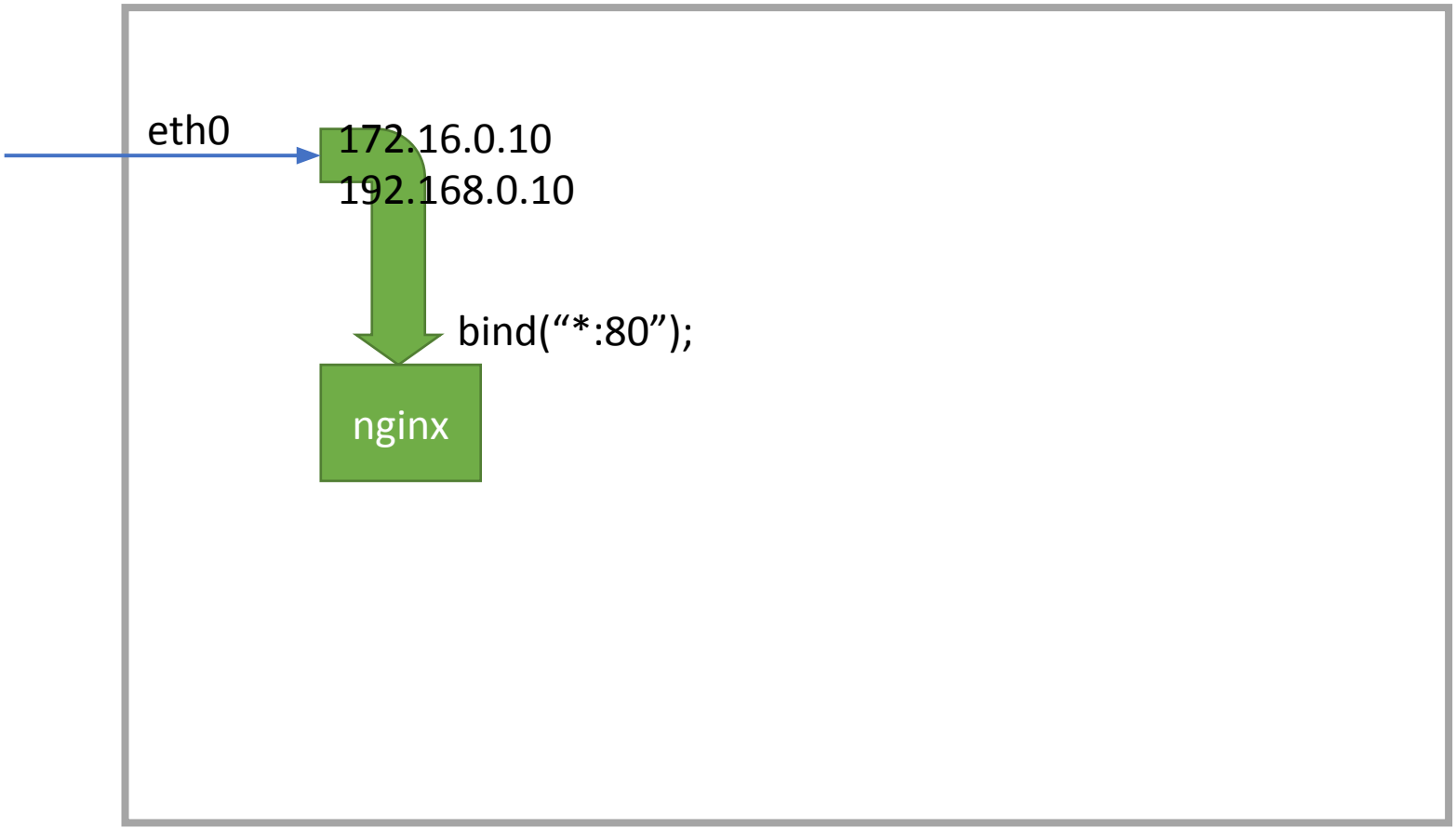
Interfaces and Bridges

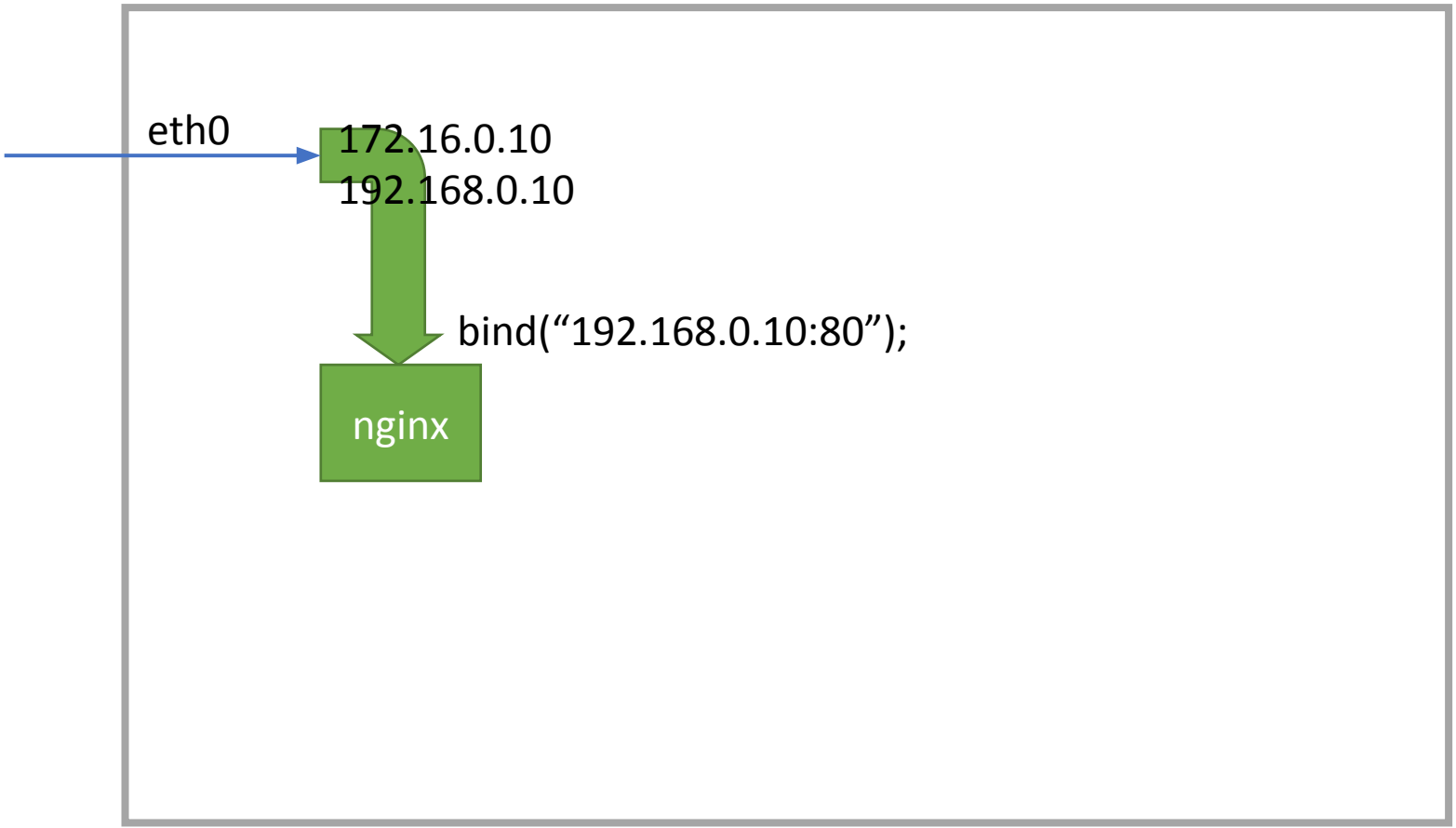

eth0
enp0s2

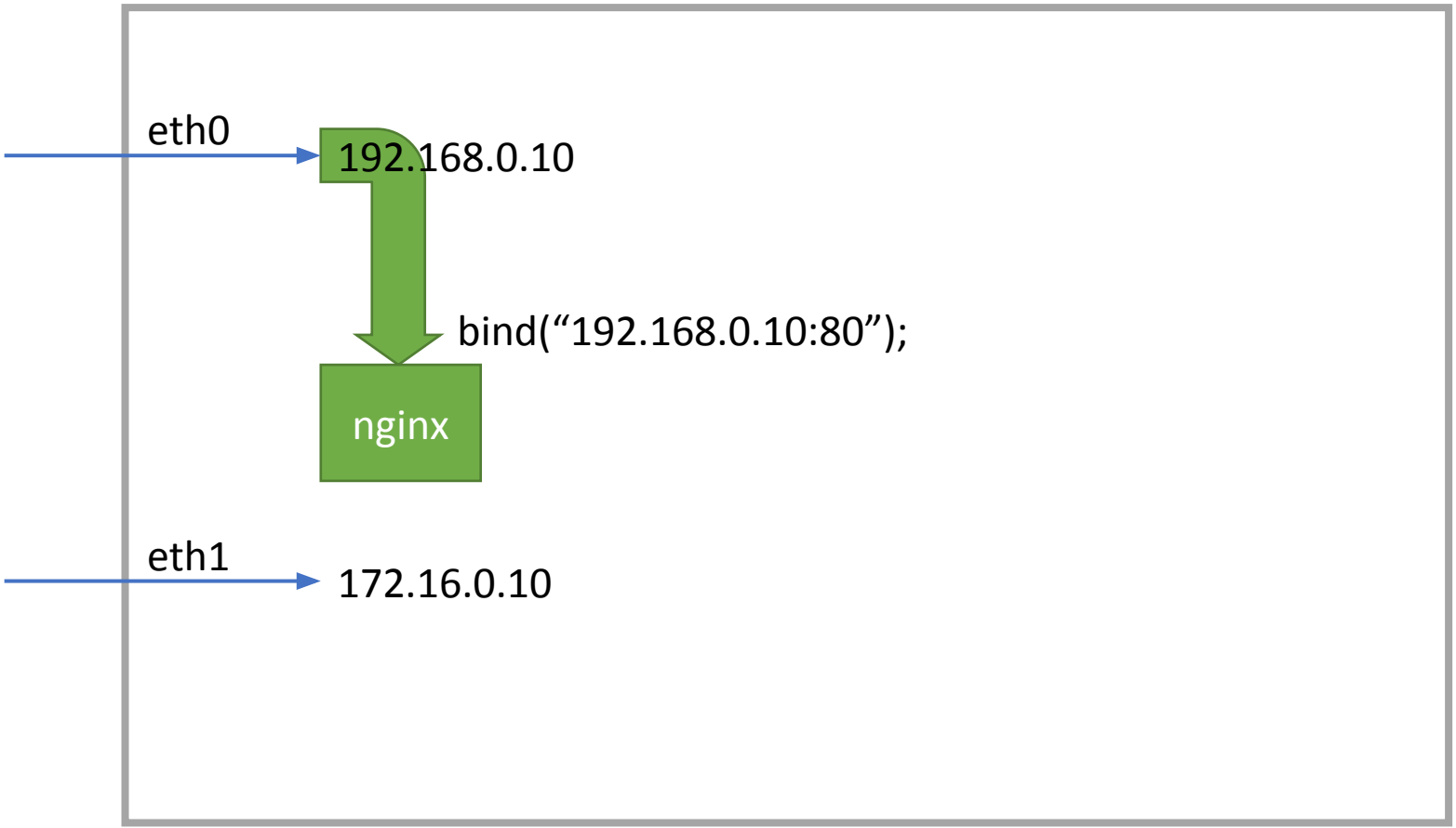

eth0
enp0s2
192.168.0.10

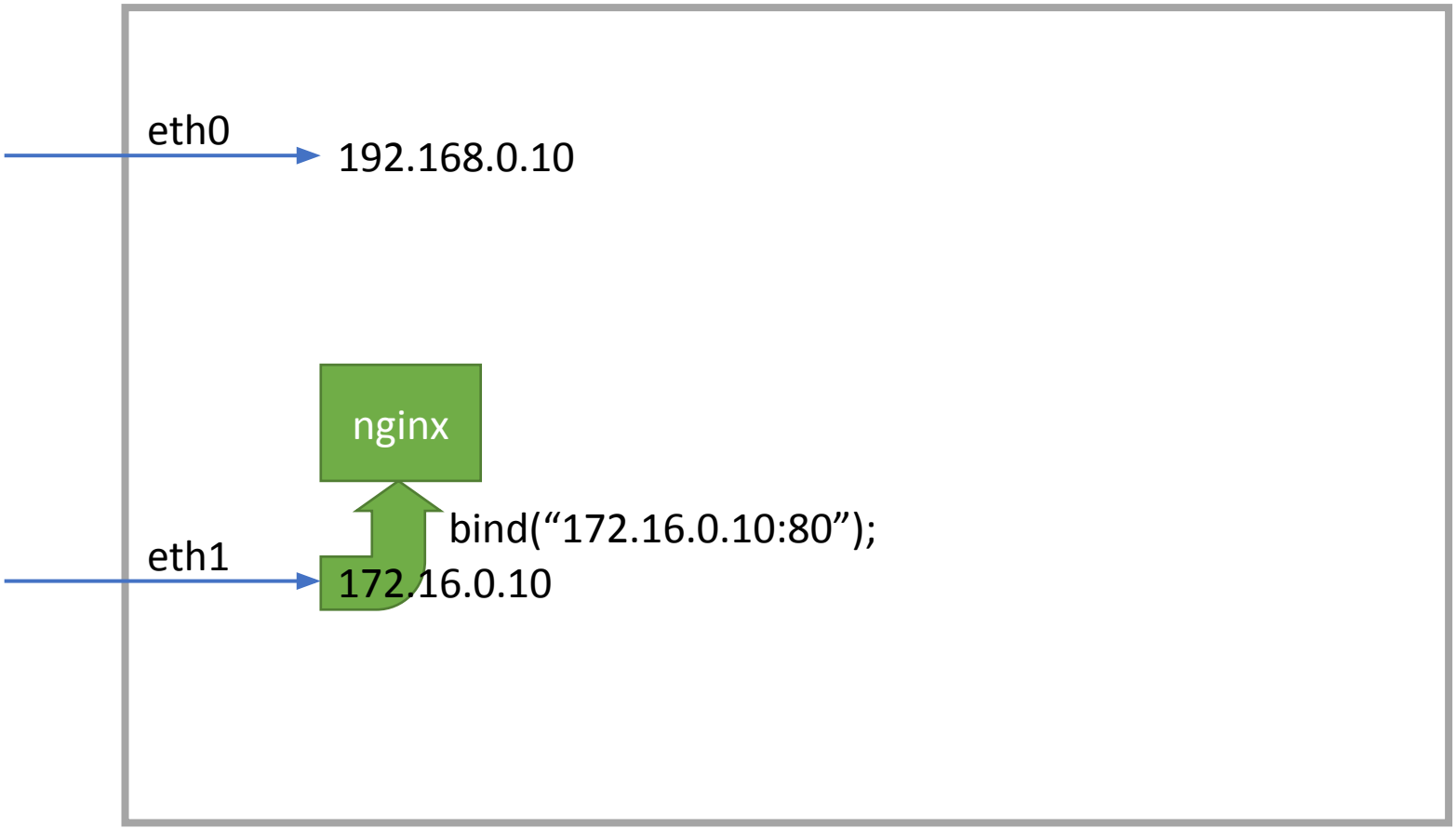

eth0
enp0s2
192.168.0.10
connect("google.com");
ping

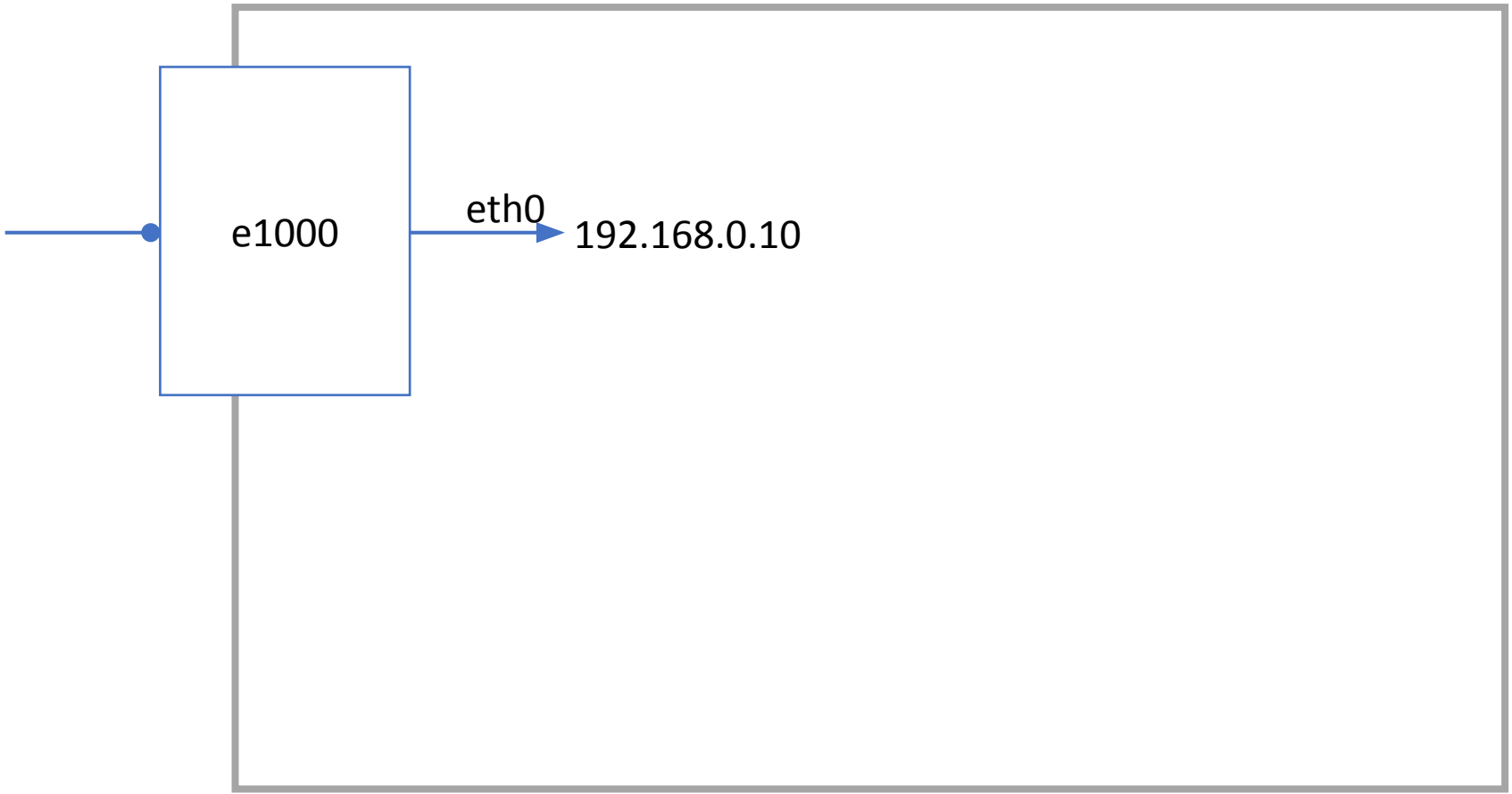

eth0
enp0s2
192.168.0.10
bind(“*:80”);
nginx

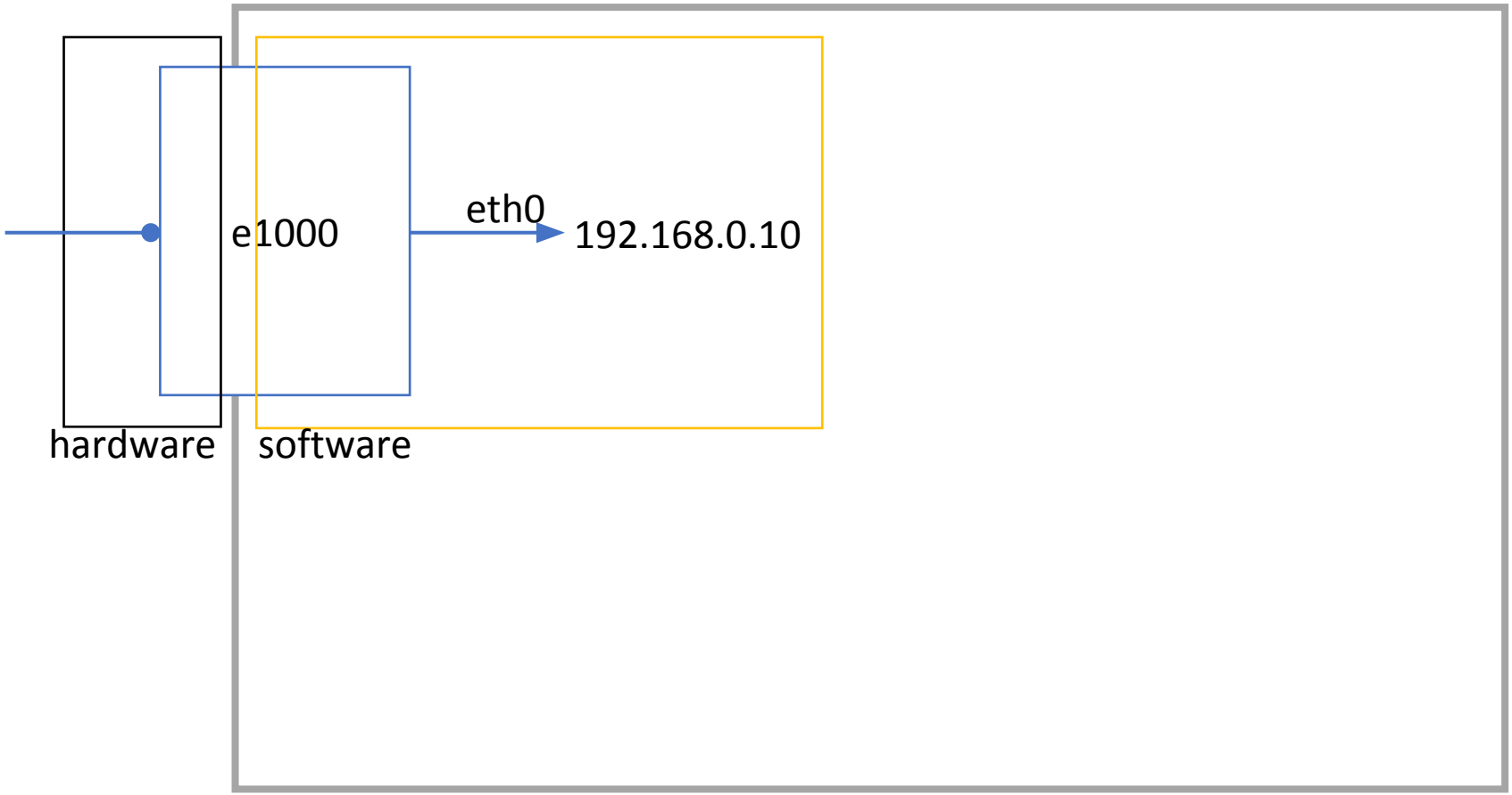

eth0
172.16.0.10
192.168.0.10
bind(“*:80”);
nginx

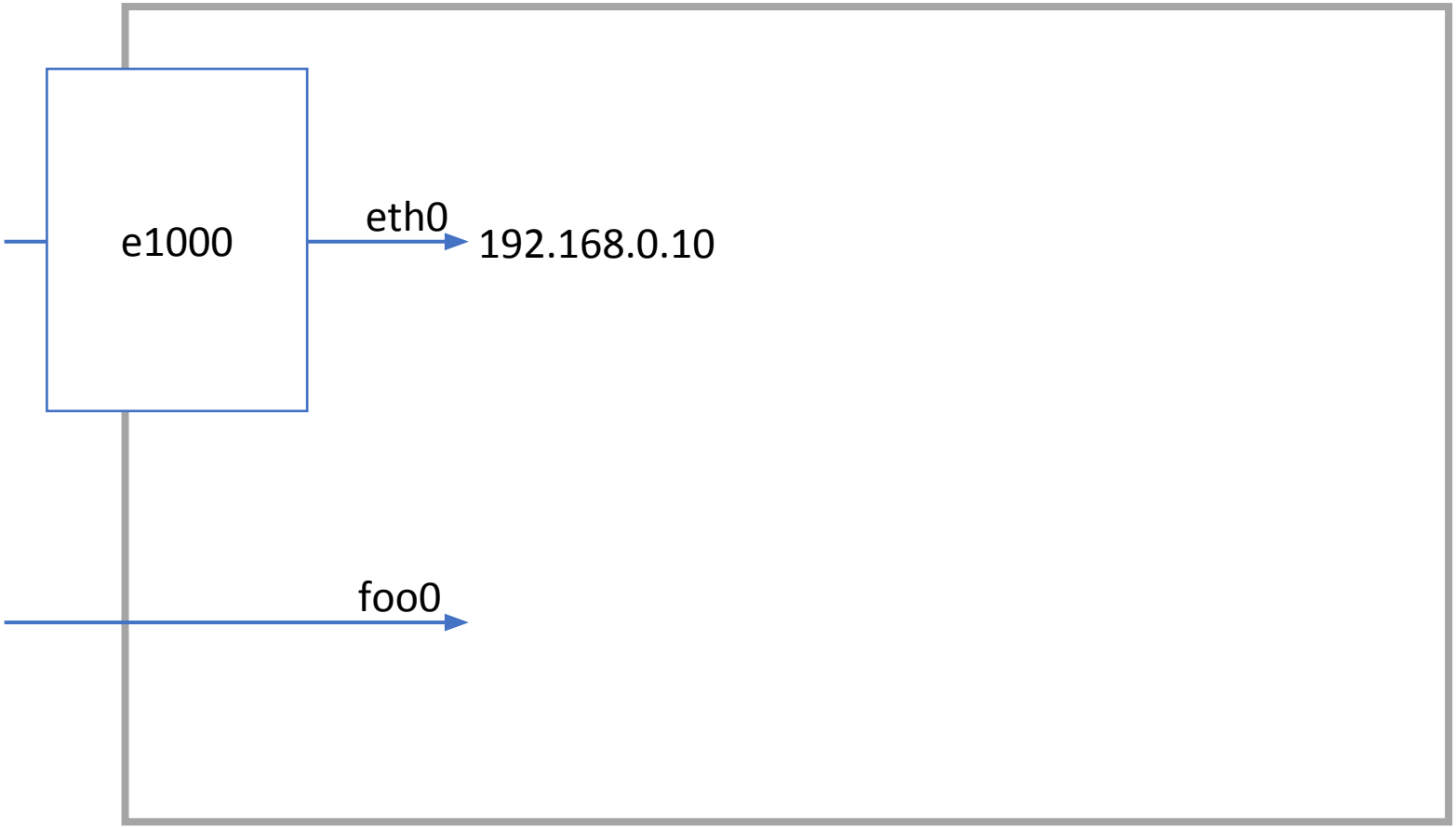

eth0
172.16.0.10
192.168.0.10
bind(“192.168.0.10:80”);
nginx

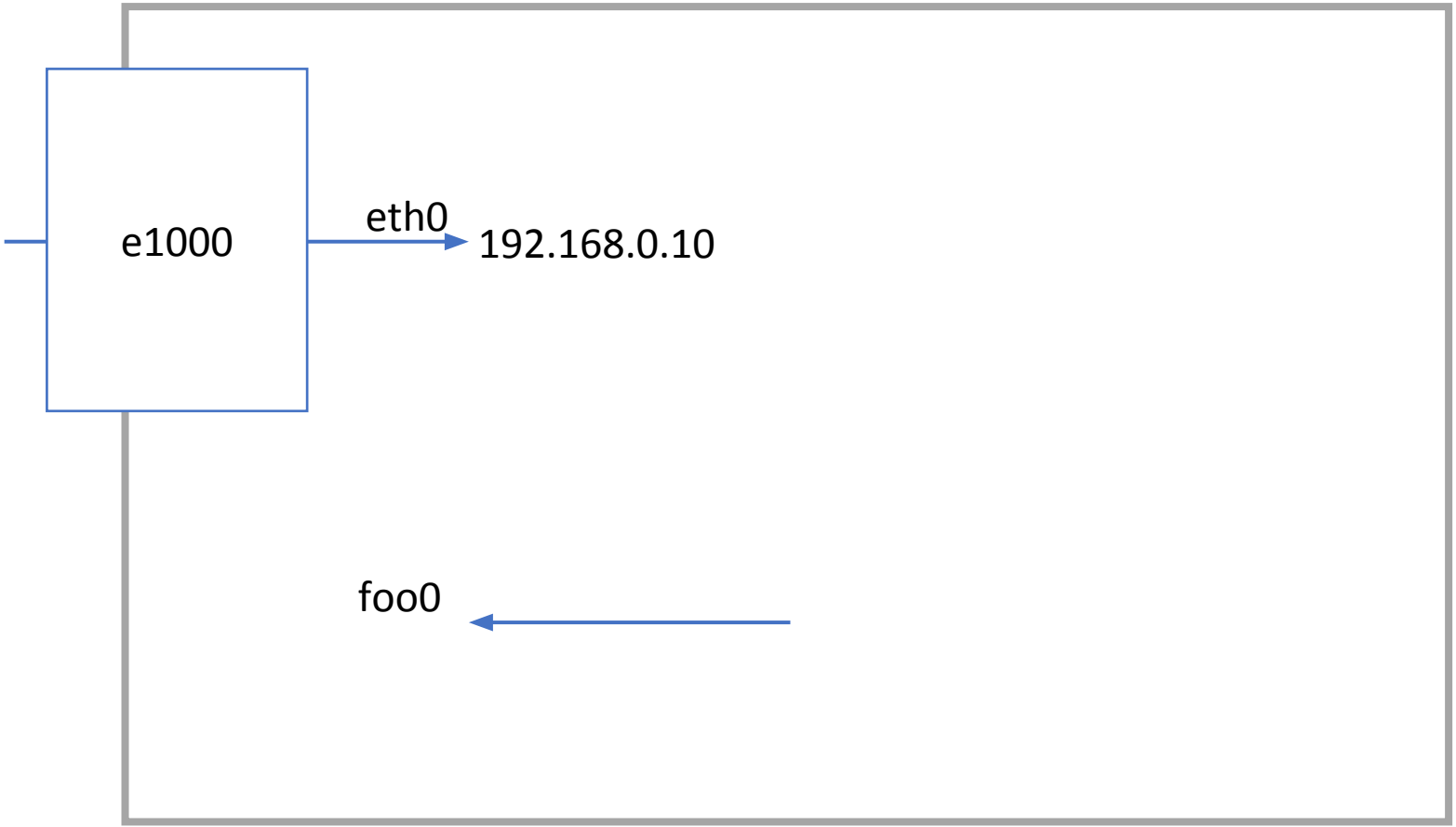

eth0
192.168.0.10
bind(“192.168.0.10:80”);
nginx
eth1
172.16.0.10

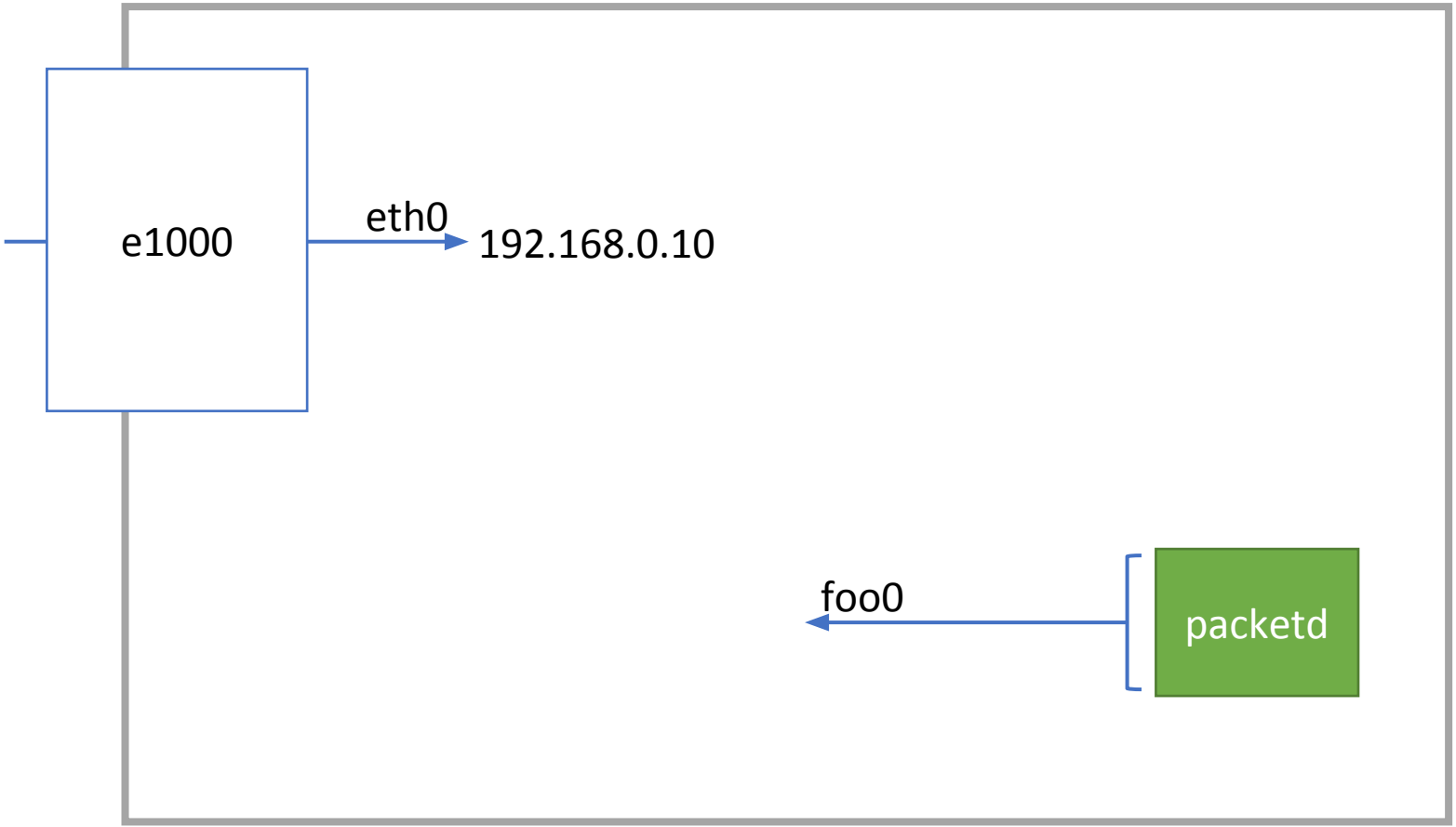

eth0
192.168.0.10
nginx
bind("172.16.0.10:80");
eth1
172.16.0.10

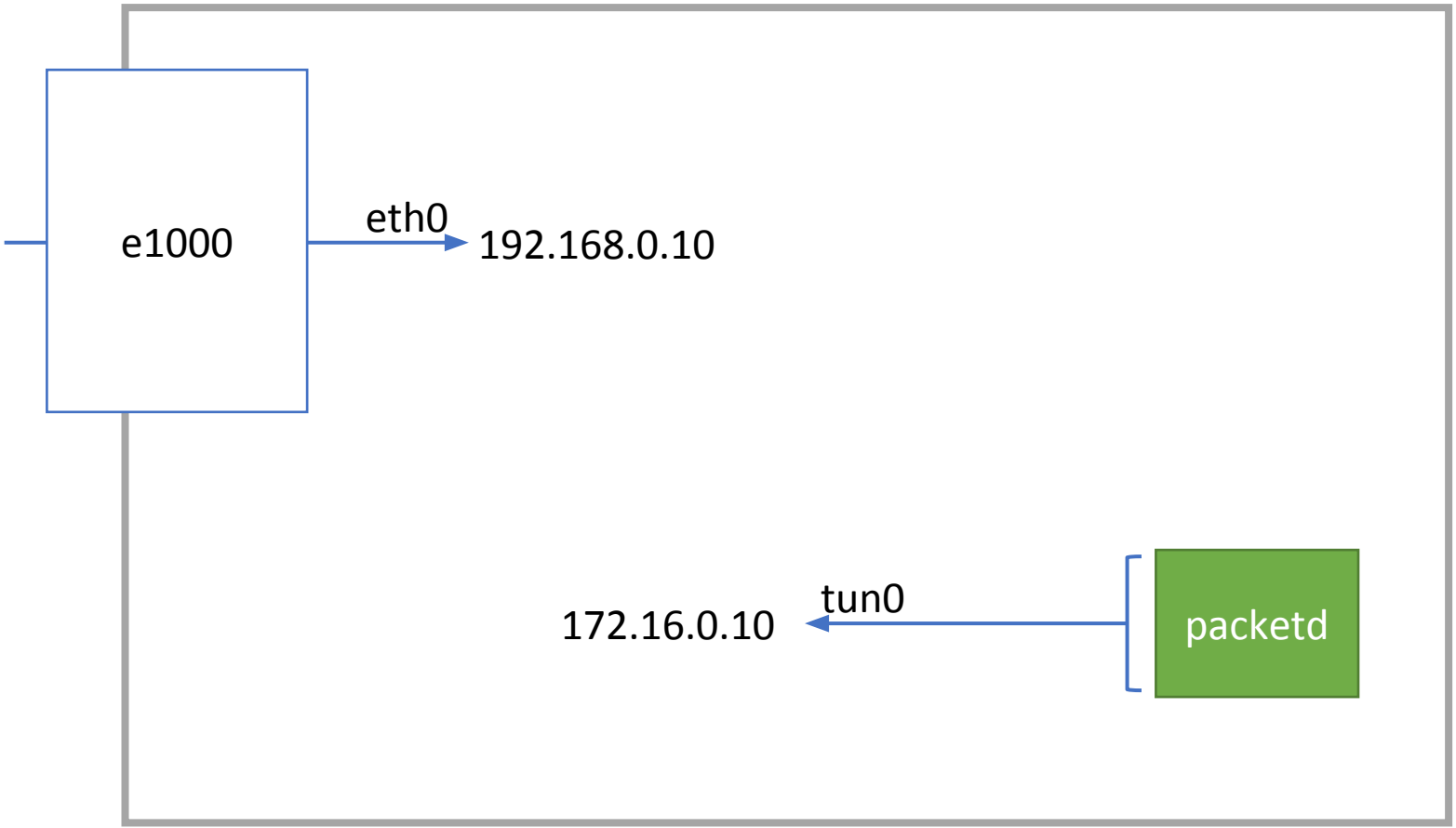

e1000
eth0
192.168.0.10

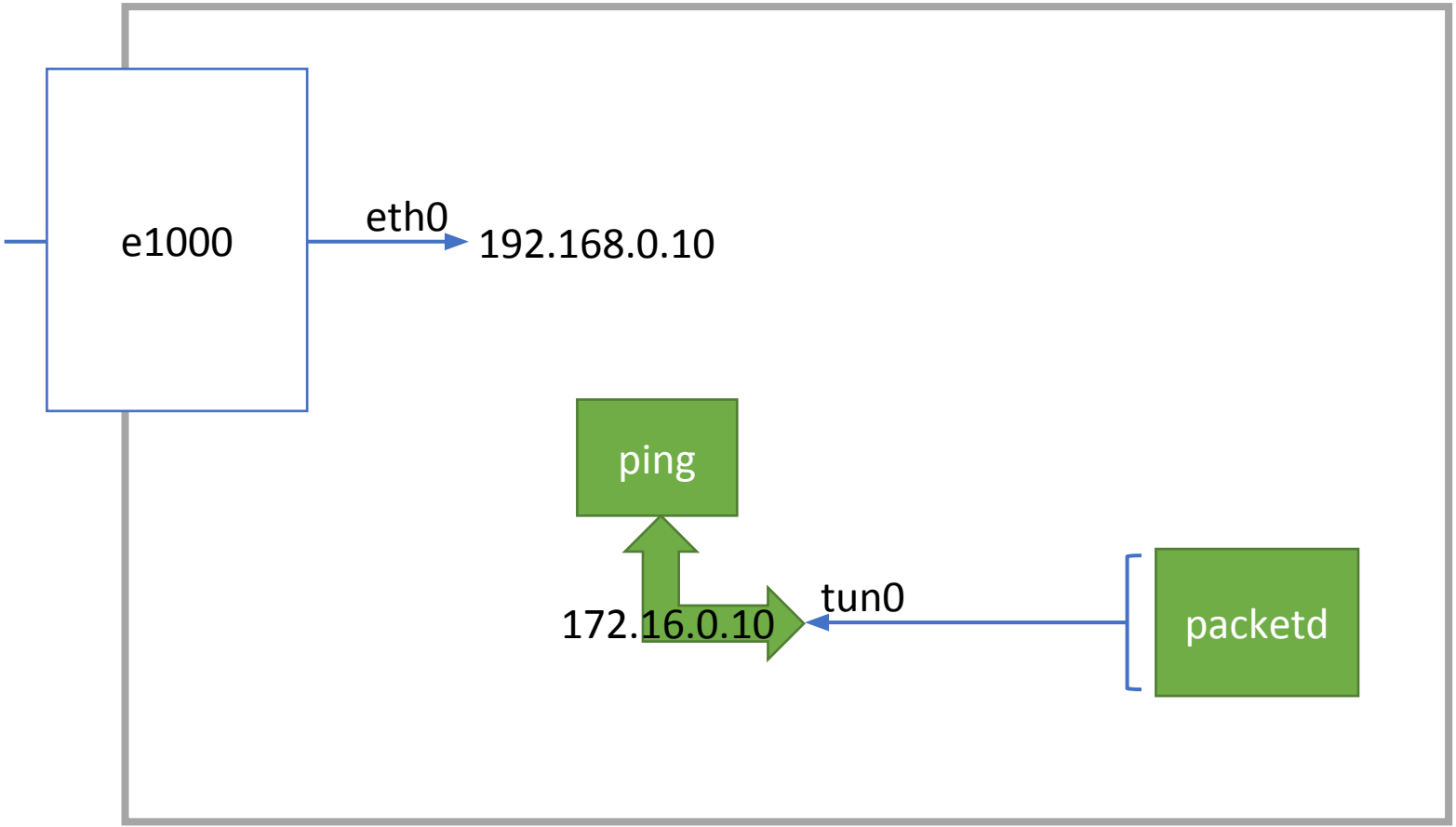

e1000
eth0
192.168.0.10
hardware
software

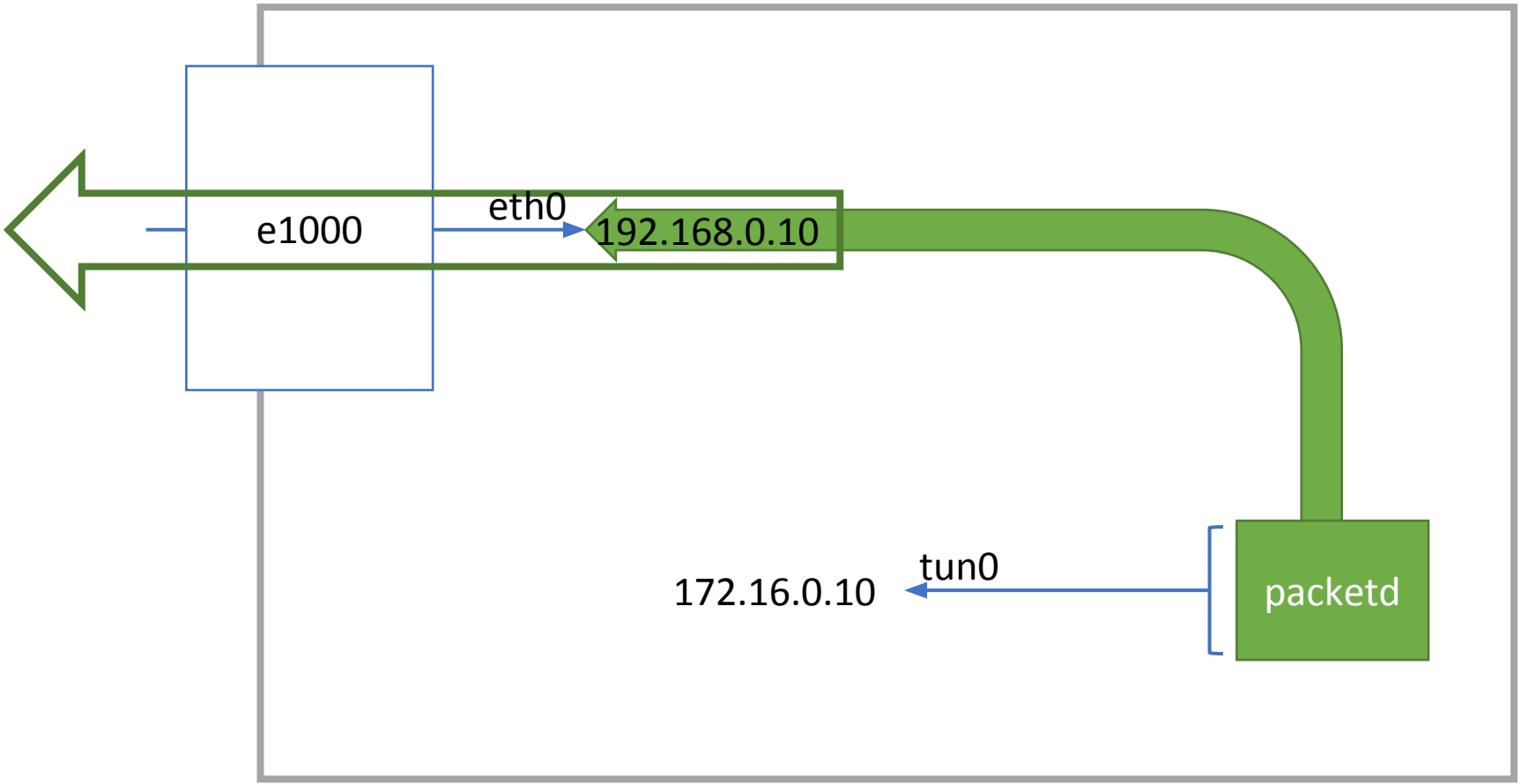

e1000
eth0
192.168.0.10
foo0

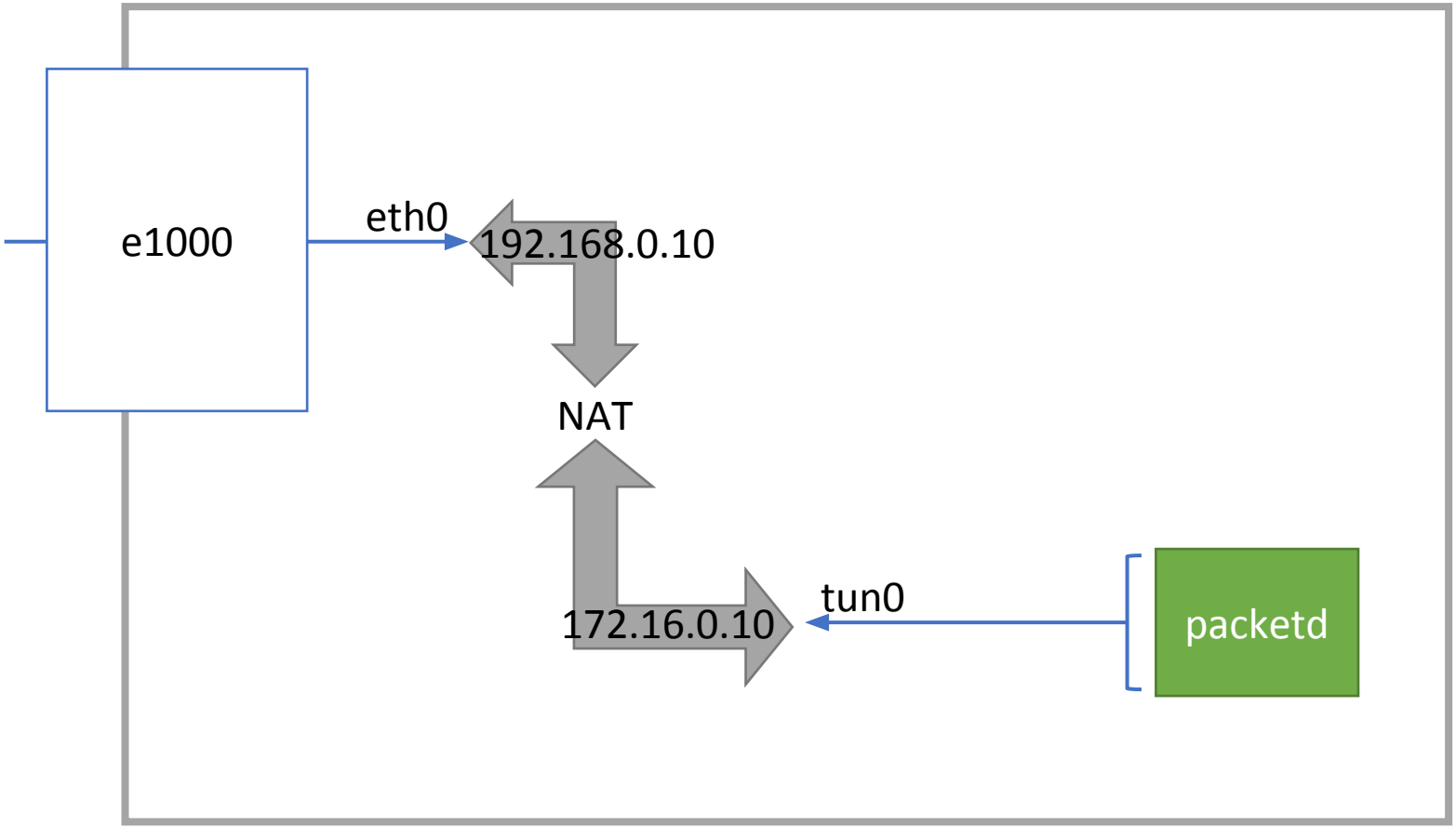

e1000
eth0
192.168.0.10
foo0

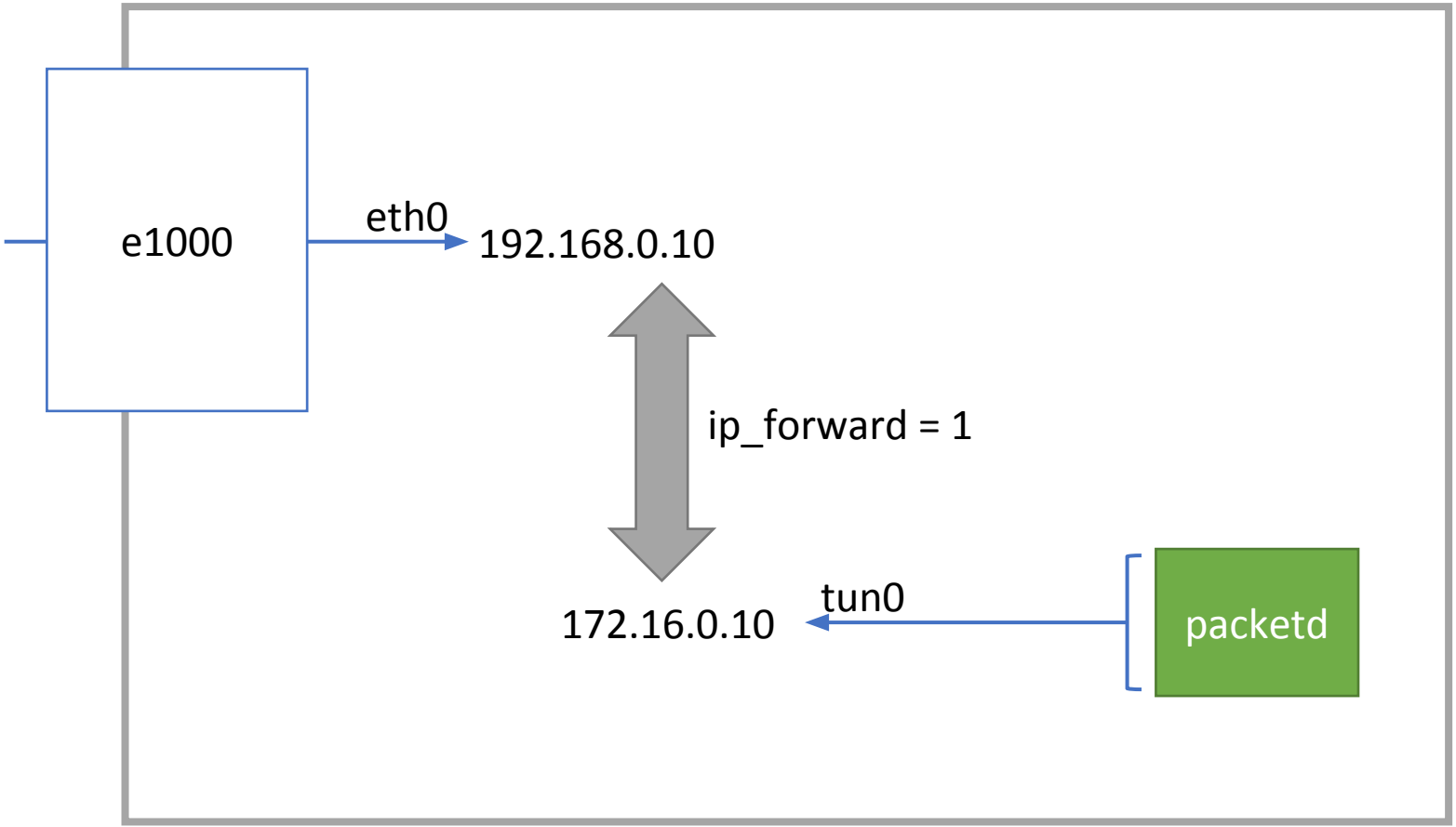

e1000
eth0
192.168.0.10
foo0
packetd

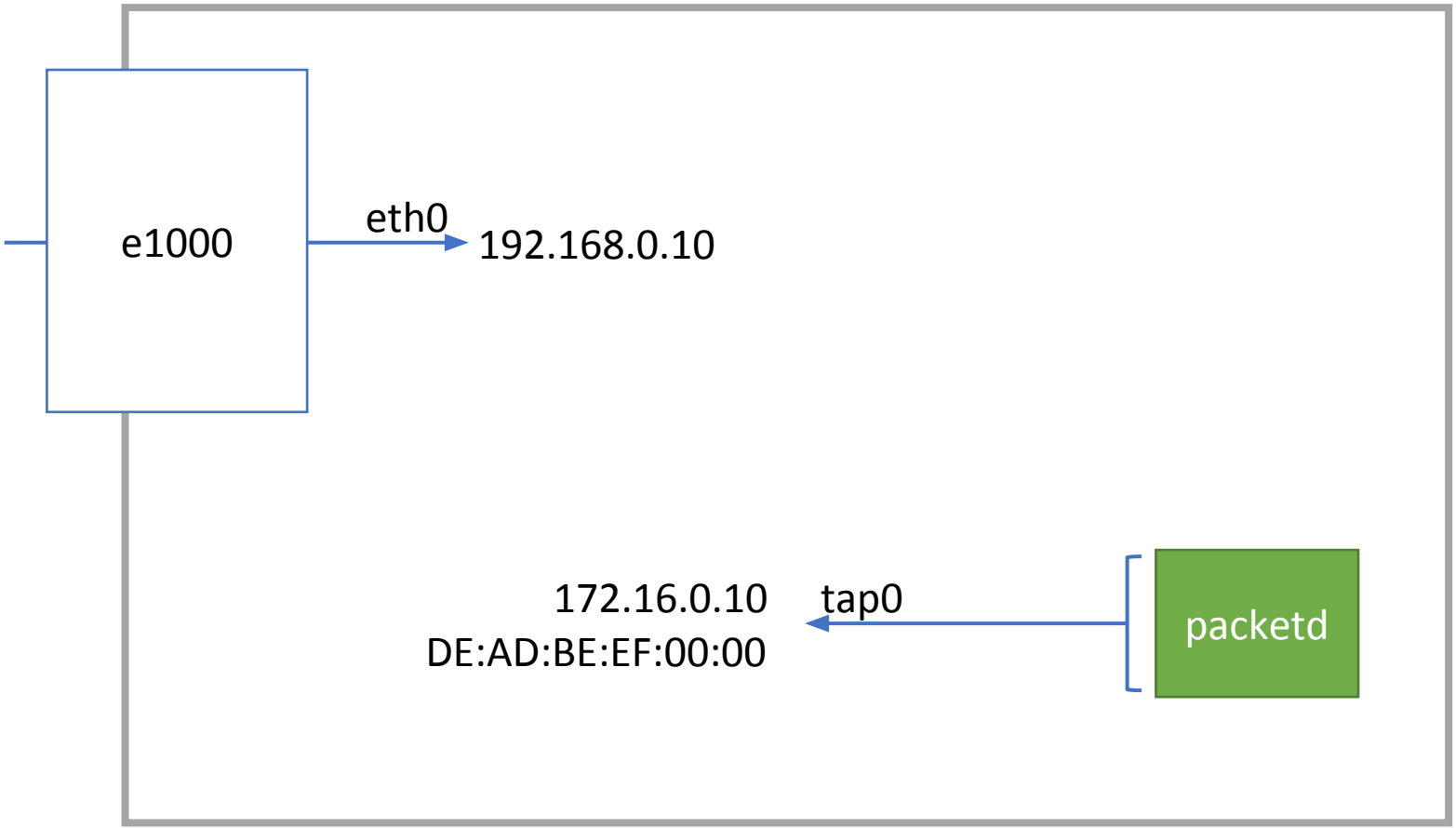

e1000
eth0
192.168.0.10
172.16.0.10
tun0
packetd

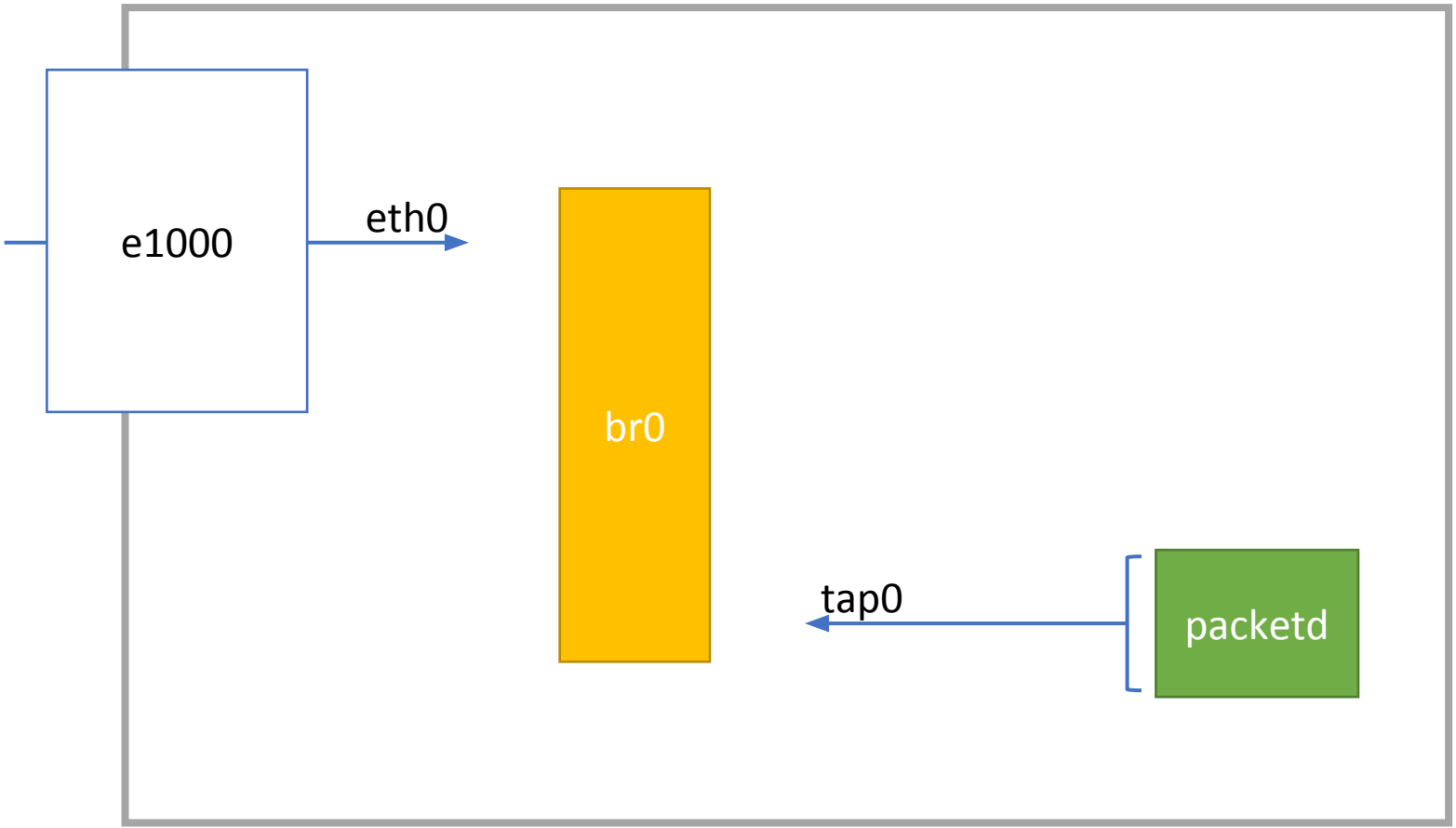

e1000
eth0
192.168.0.10
ping
172.16.0.10
tun0
packetd

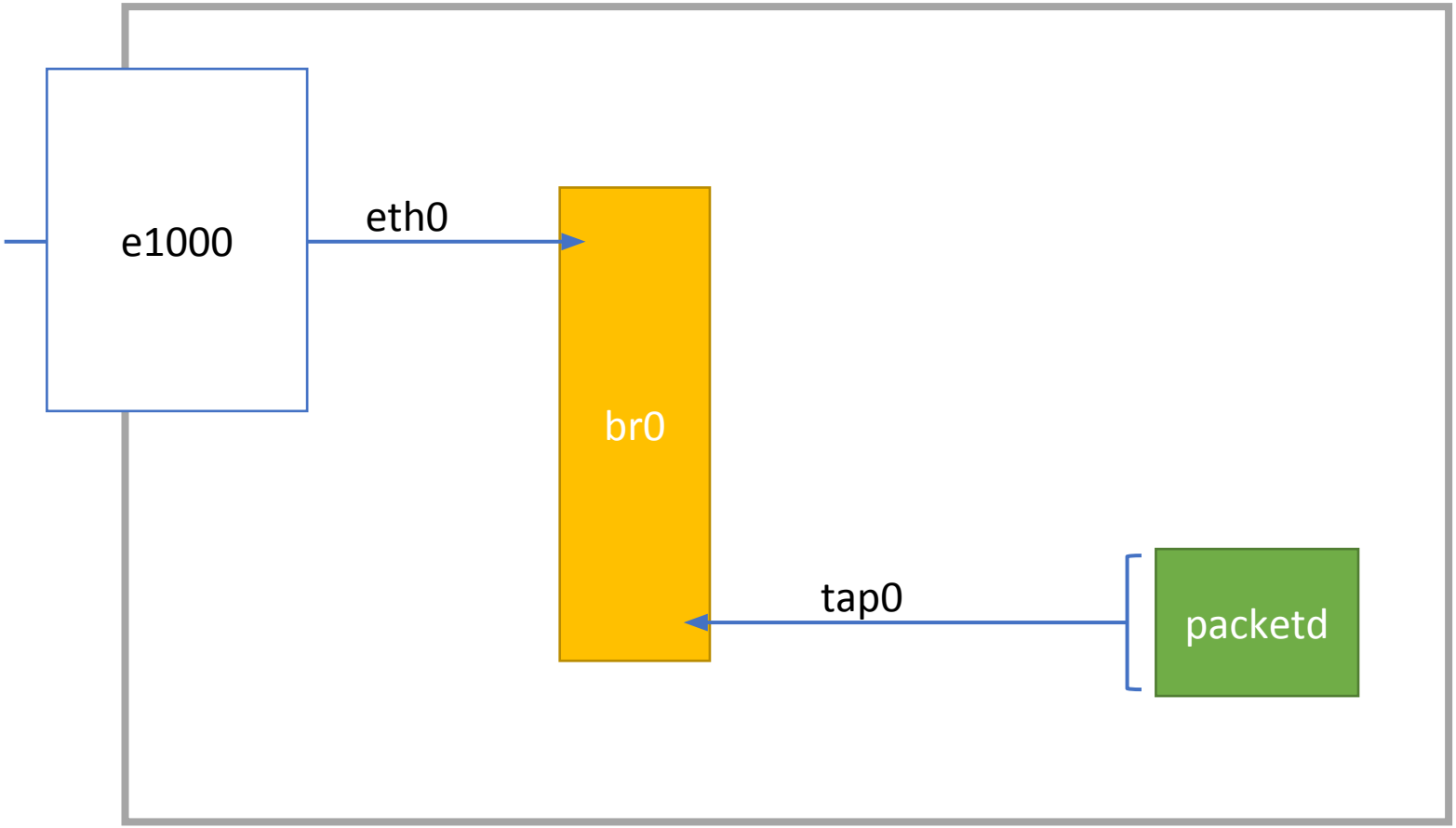

e1000
eth0
192.168.0.10
172.16.0.10
tun0
packetd

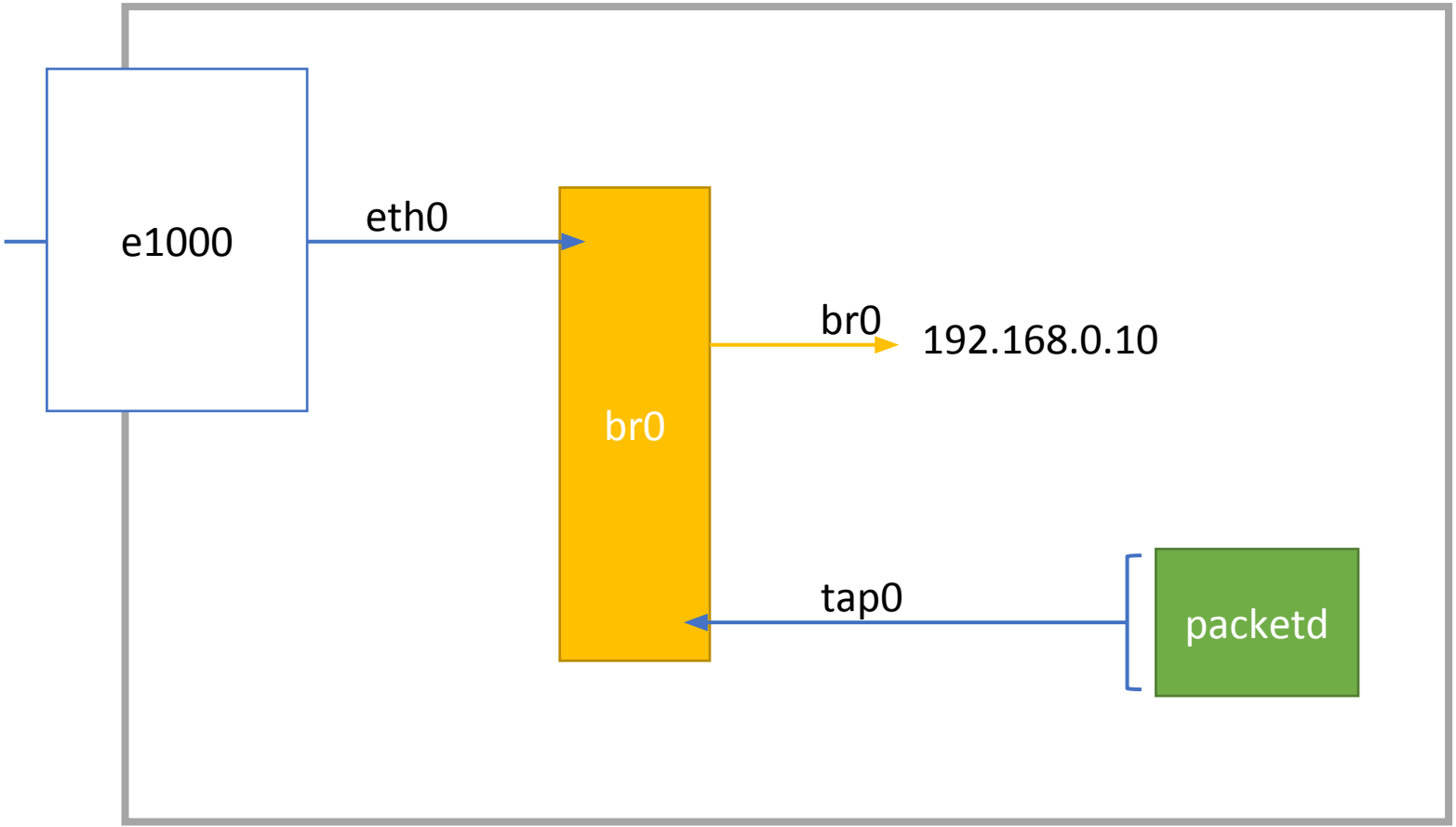

e1000
eth0
192.168.0.10
NAT
172.16.0.10
tun0
packetd

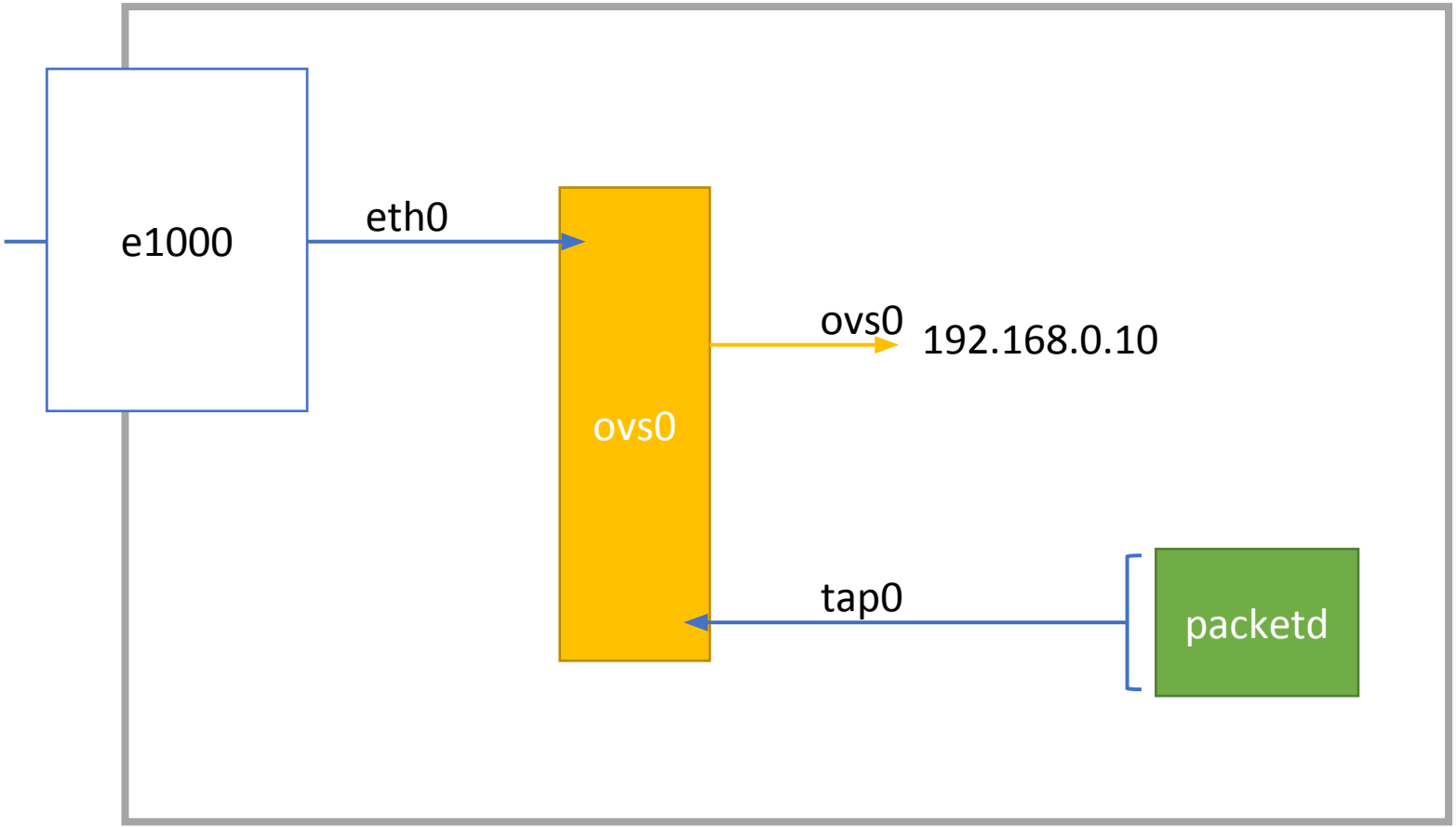

e1000
eth0
192.168.0.10
ip_forward = 1
172.16.0.10
tun0
packetd

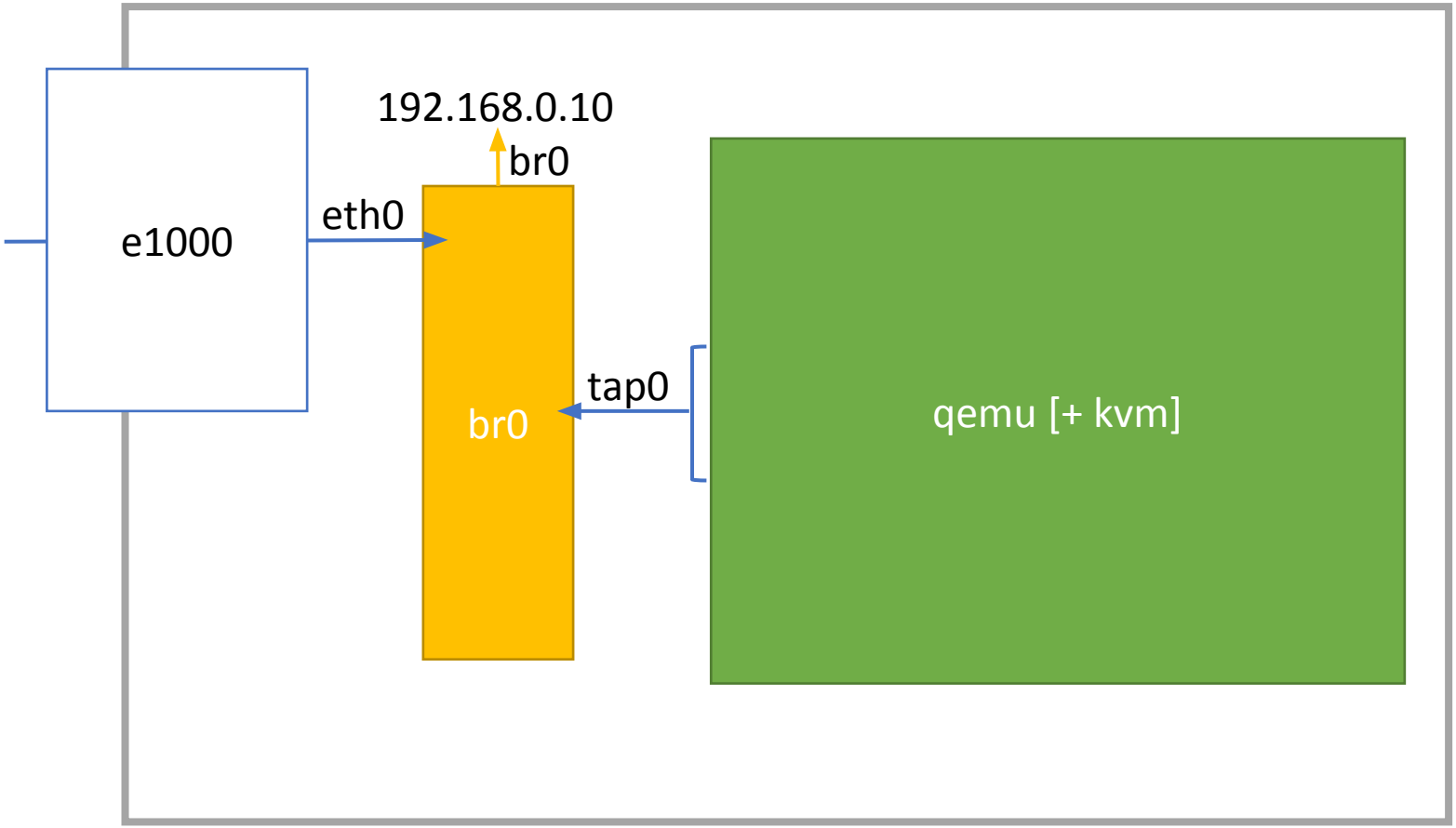

e1000
eth0
192.168.0.10
172.16.0.10
DE:AD:BE:EF:00:00
tap0
packetd

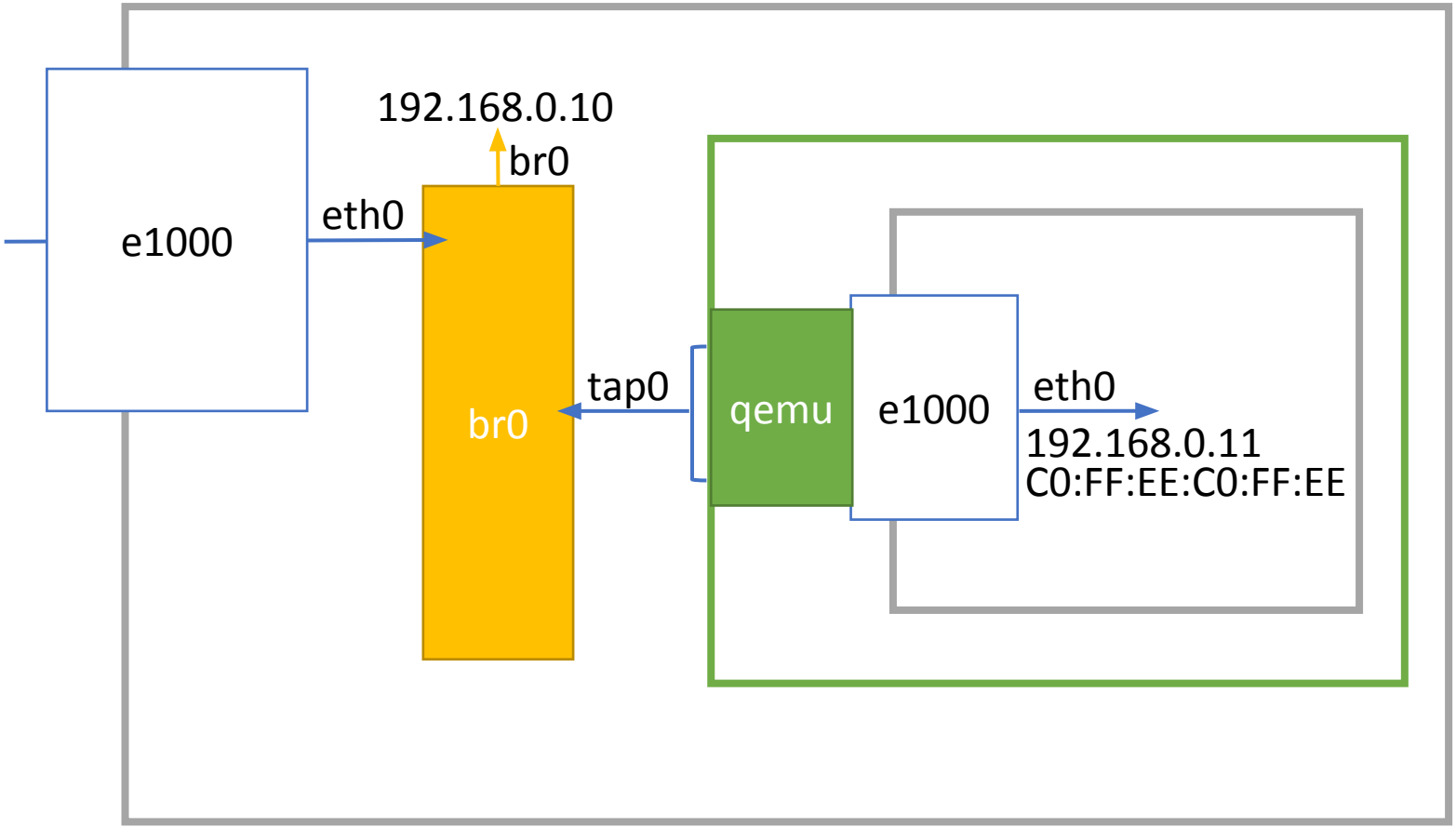

e1000
eth0
br0
tap0
packetd

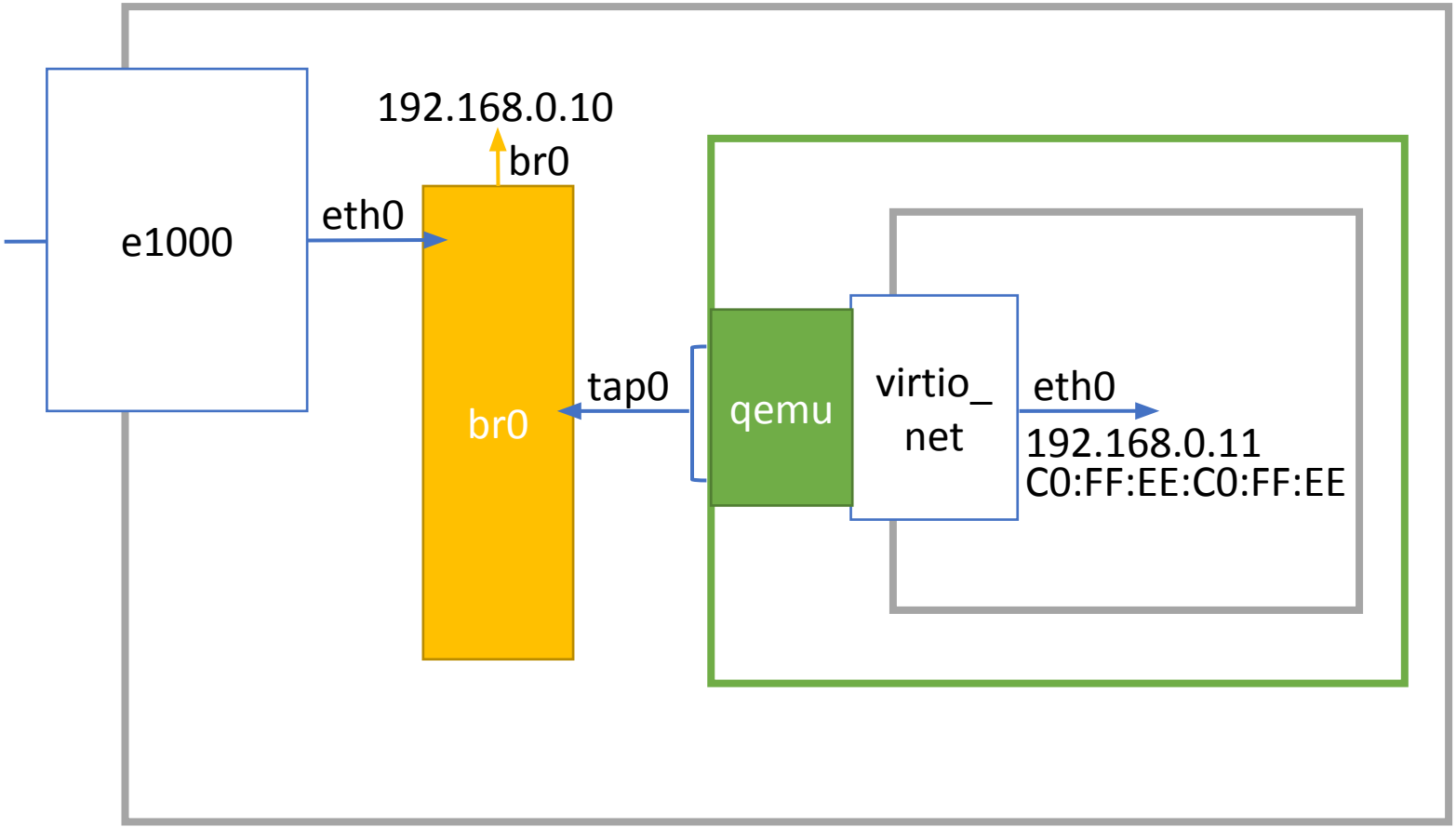

e1000
eth0
br0
tap0
packetd

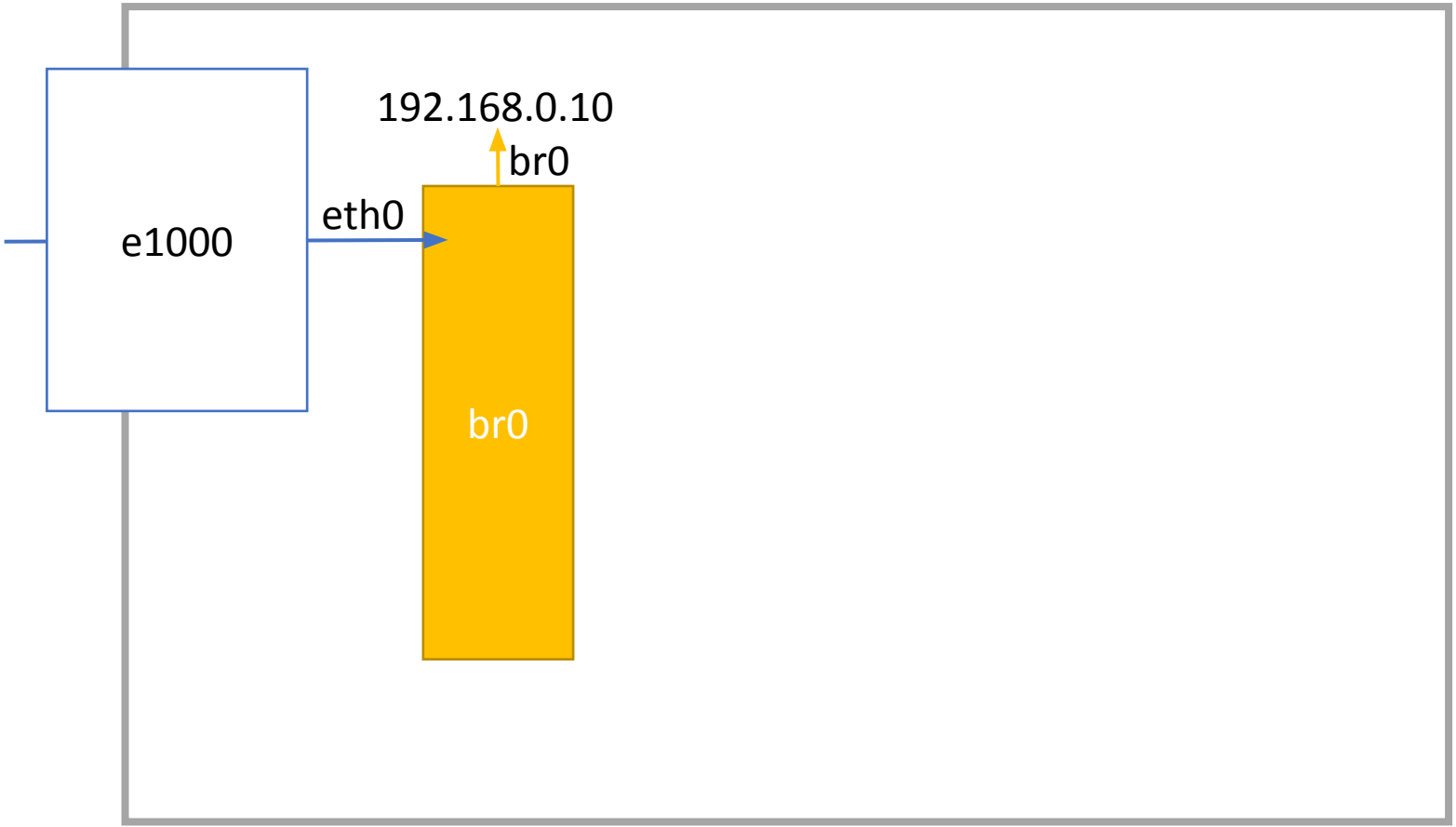

e1000
eth0
br0
br0
192.168.0.10
tap0
packetd

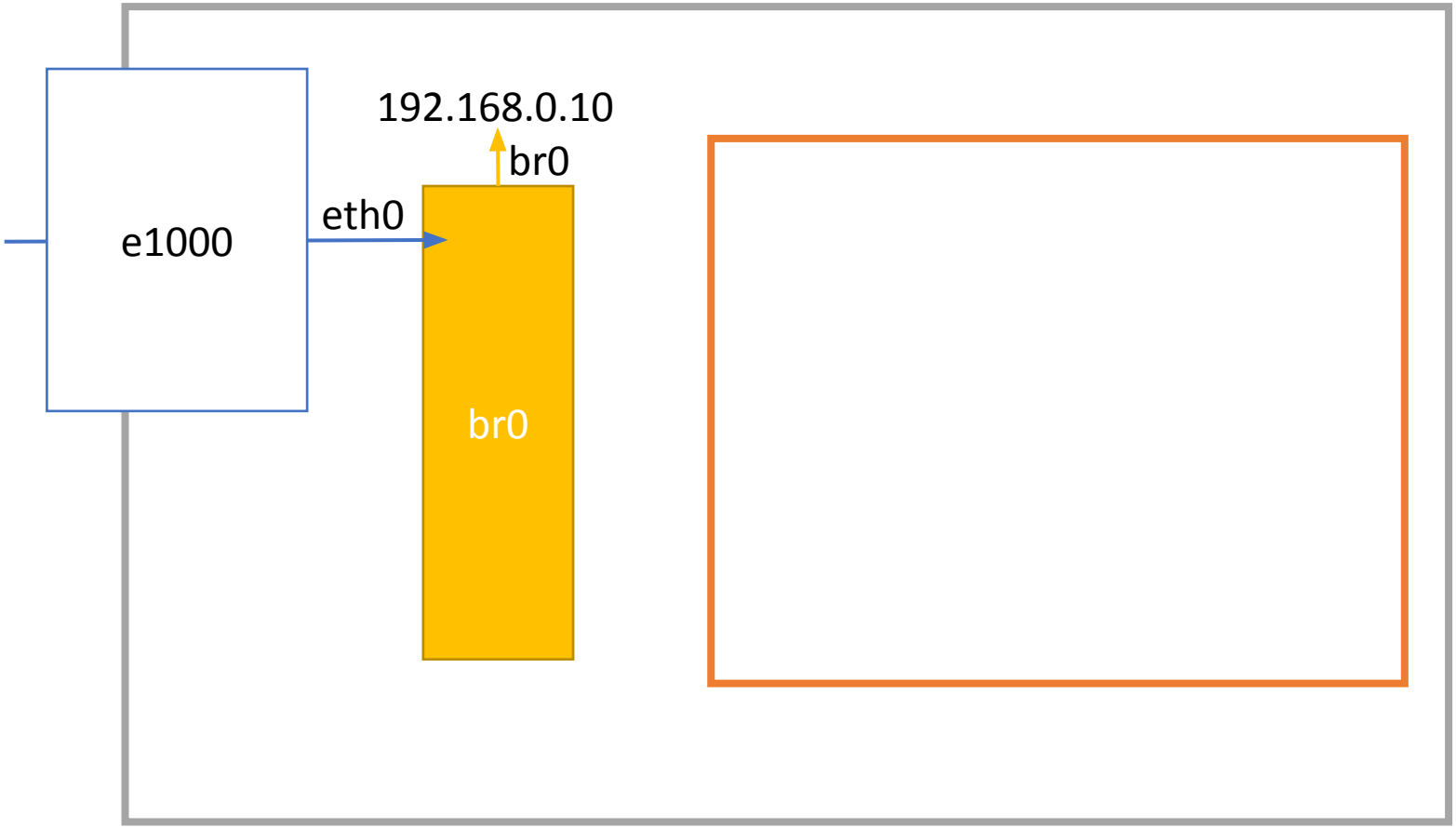

e1000
eth0
ovs0
ovs0
192.168.0.10
tap0
packetd

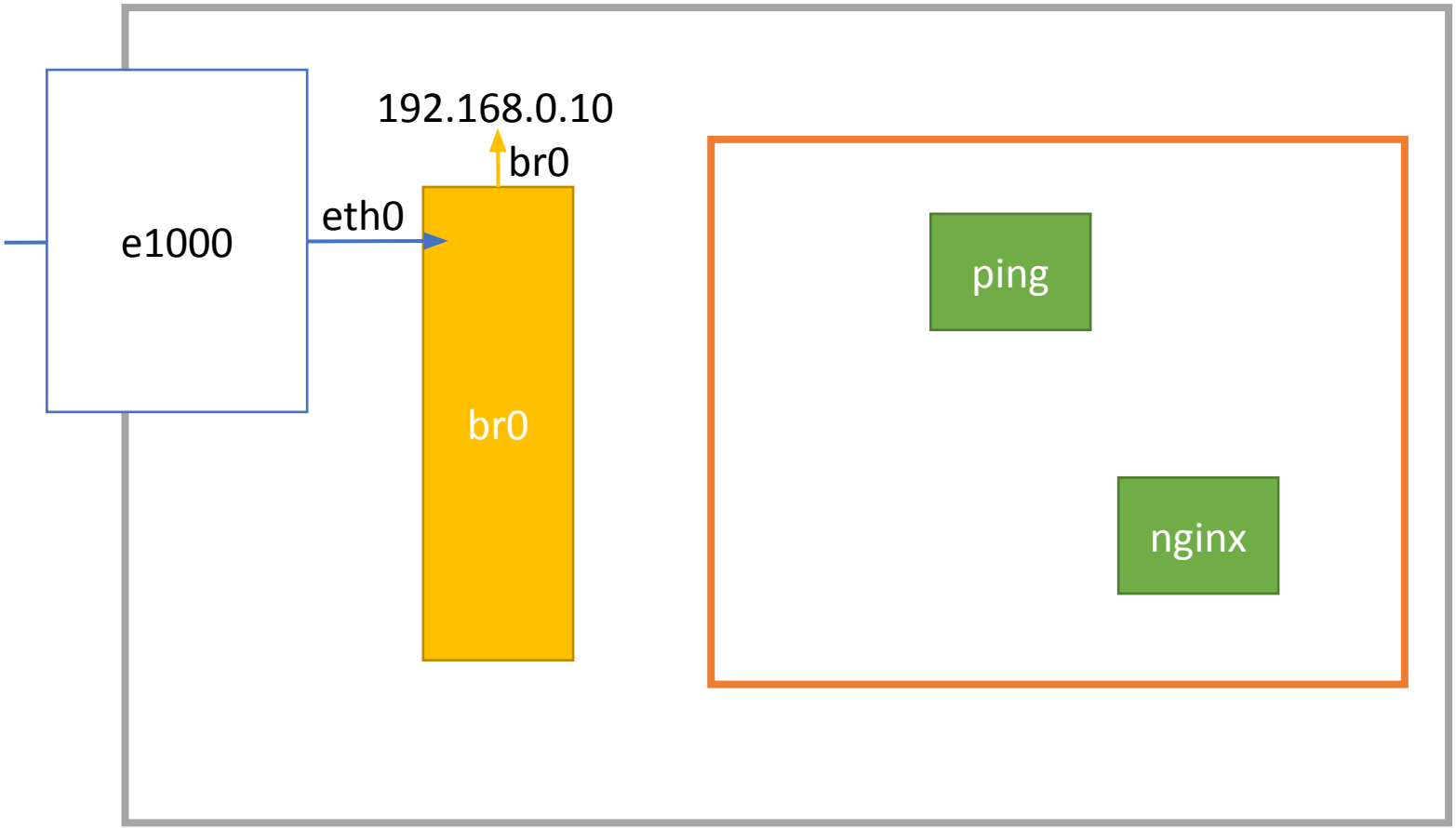

e1000
eth0
192.168.0.10
br0
br0
tap0
qemu [+ kvm]

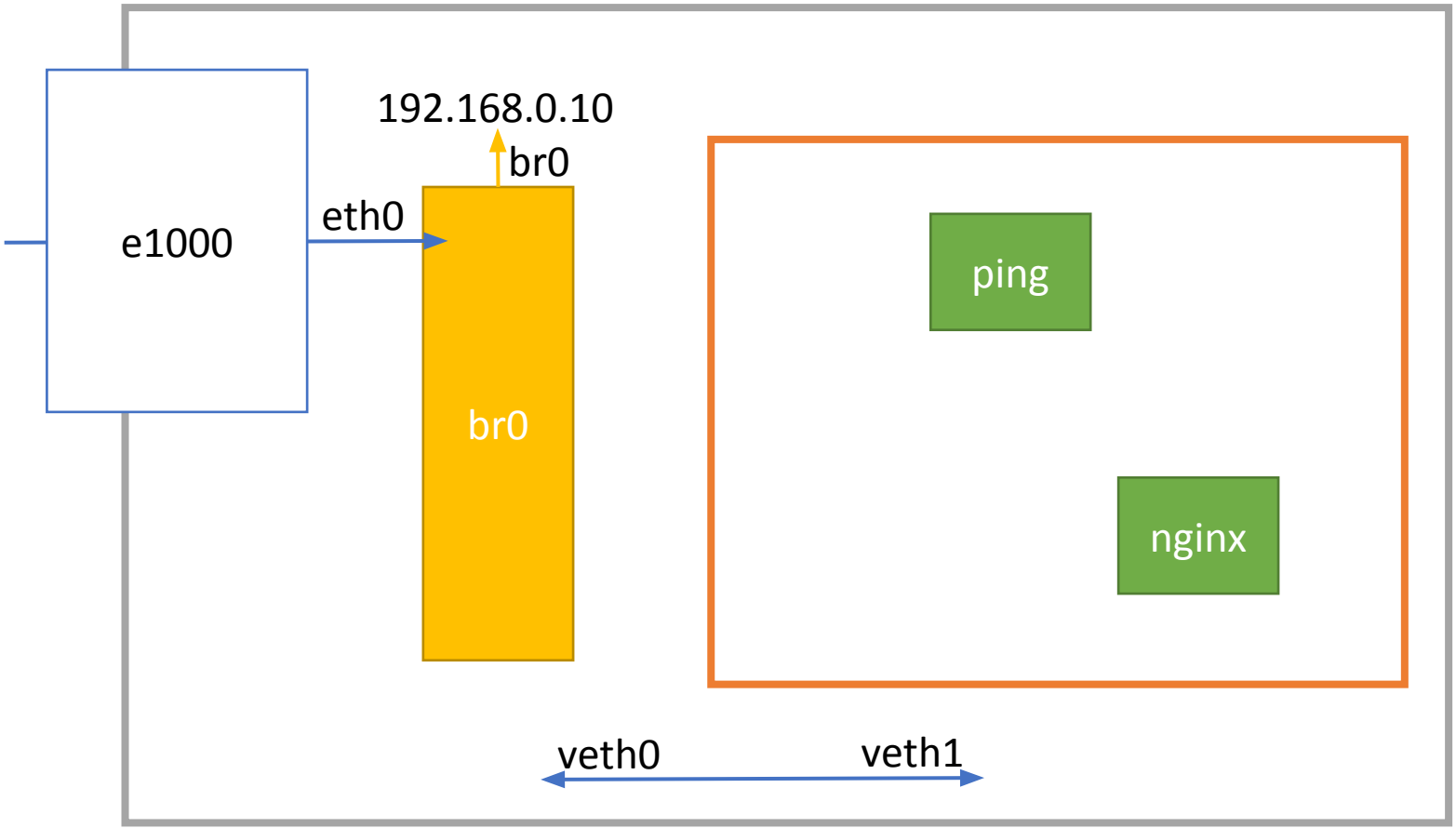

e1000
192.168.0.10
br0
eth0
br0
tap0
qemu
e1000
eth0
192.168.0.11
C0:FF:EE:C0:FF:EE

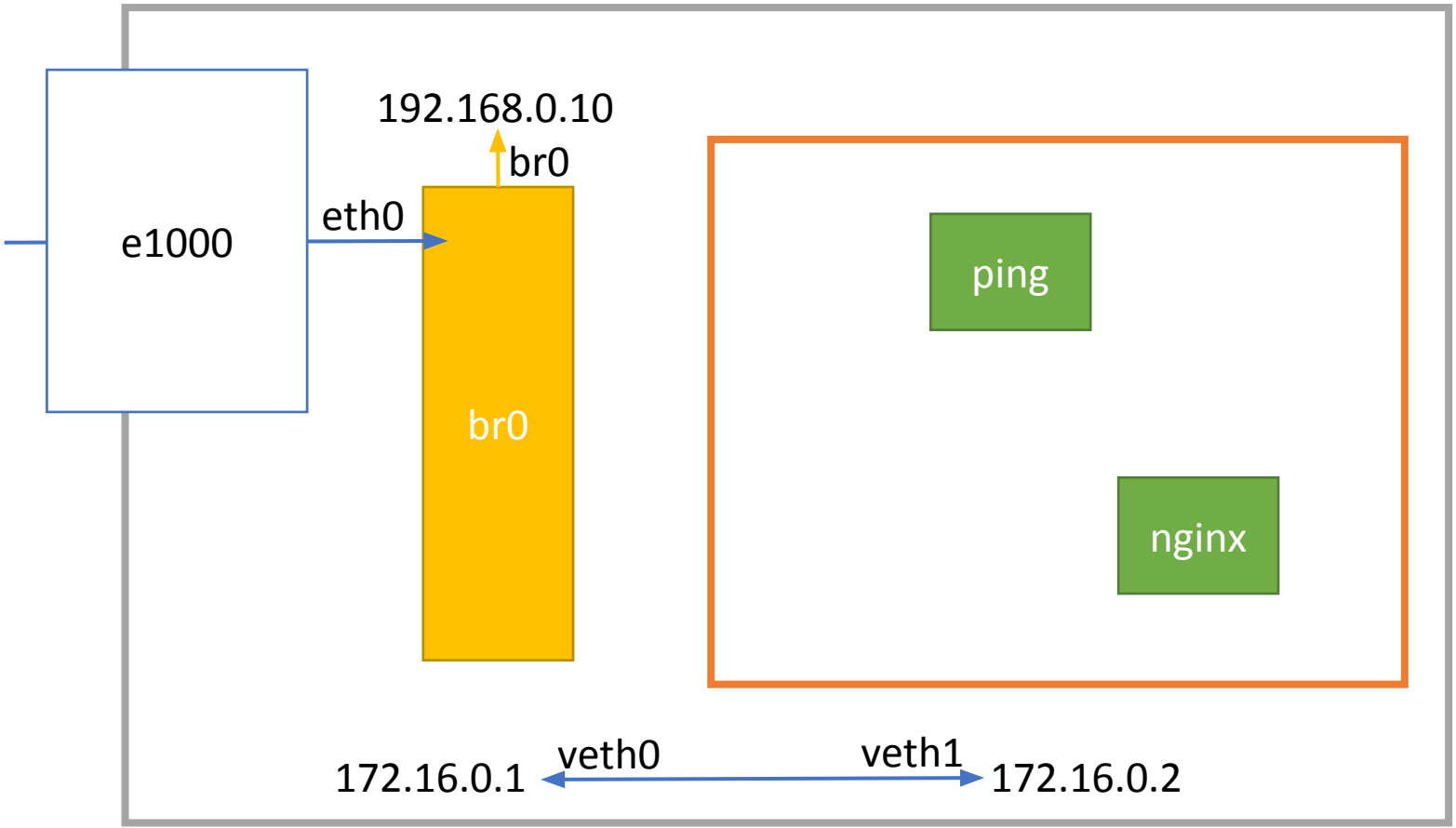

e1000
eth0
192.168.0.10
br0
br0
tap0
qemu
virtio_
net
eth0
192.168.0.11
C0:FF:EE:C0:FF:EE

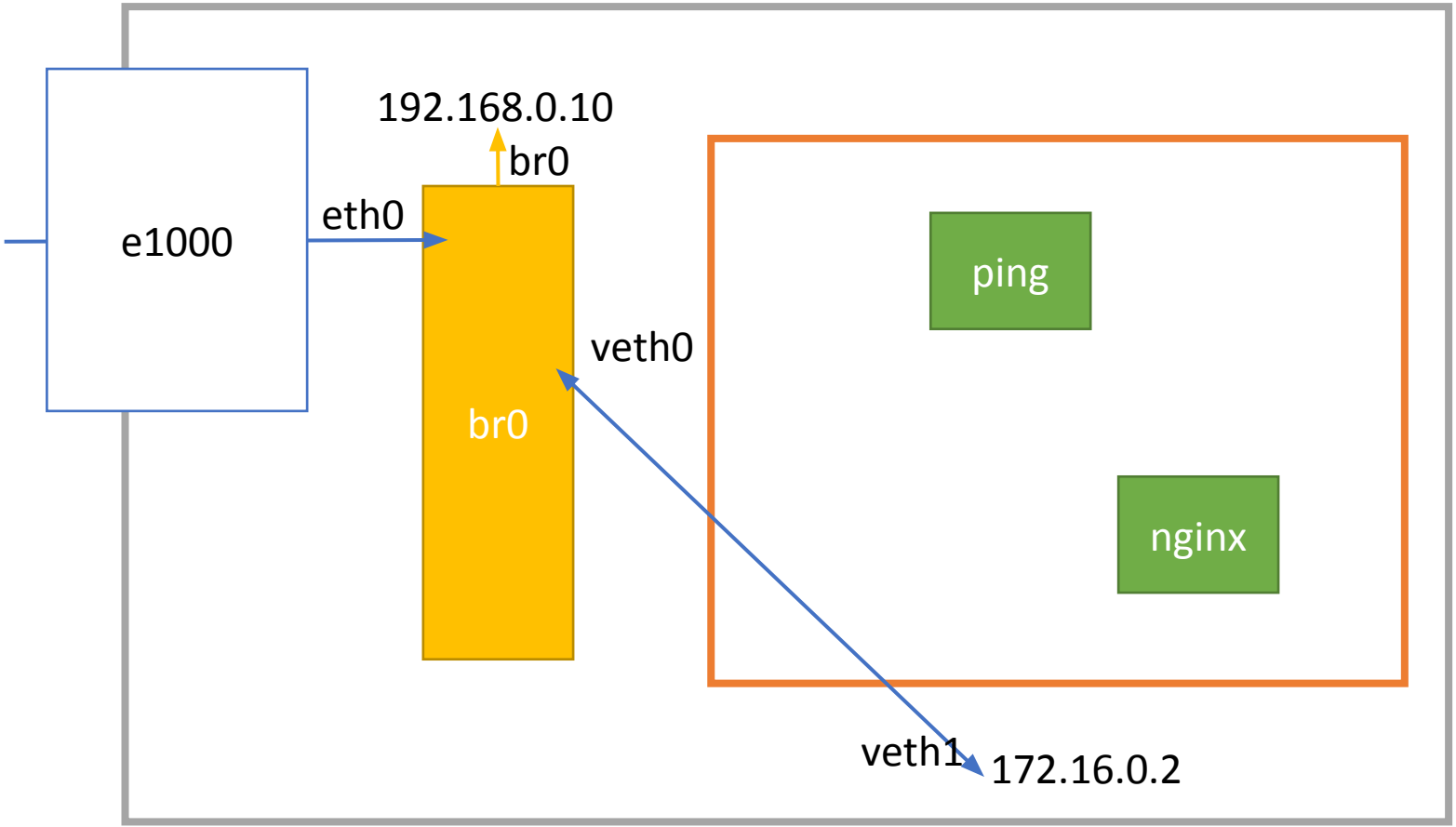

192.168.0.10
br0
e1000
eth0
br0

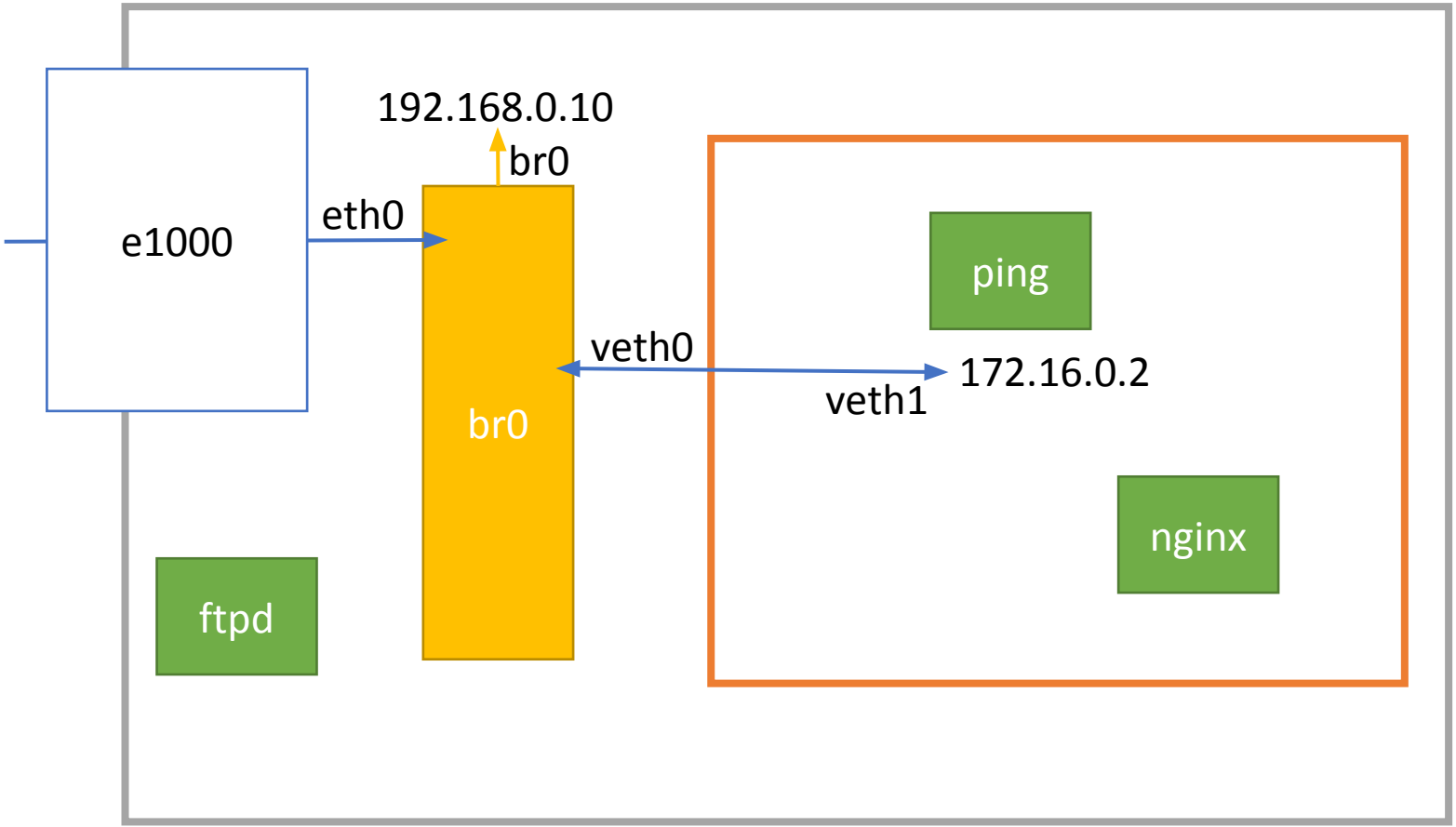

192.168.0.10
br0
e1000
eth0
br0

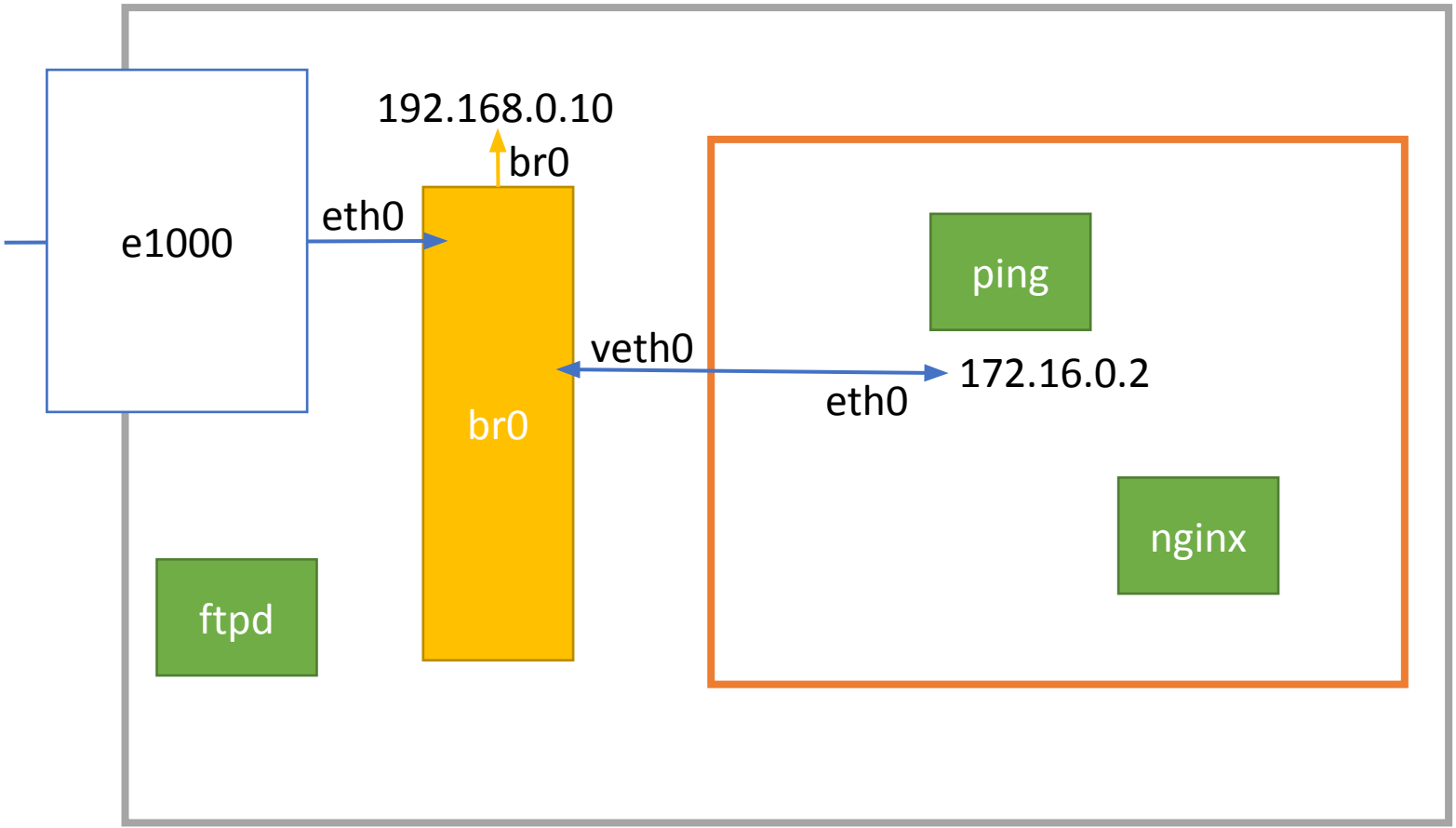

e1000
eth0
192.168.0.10
br0
br0
ping
nginx

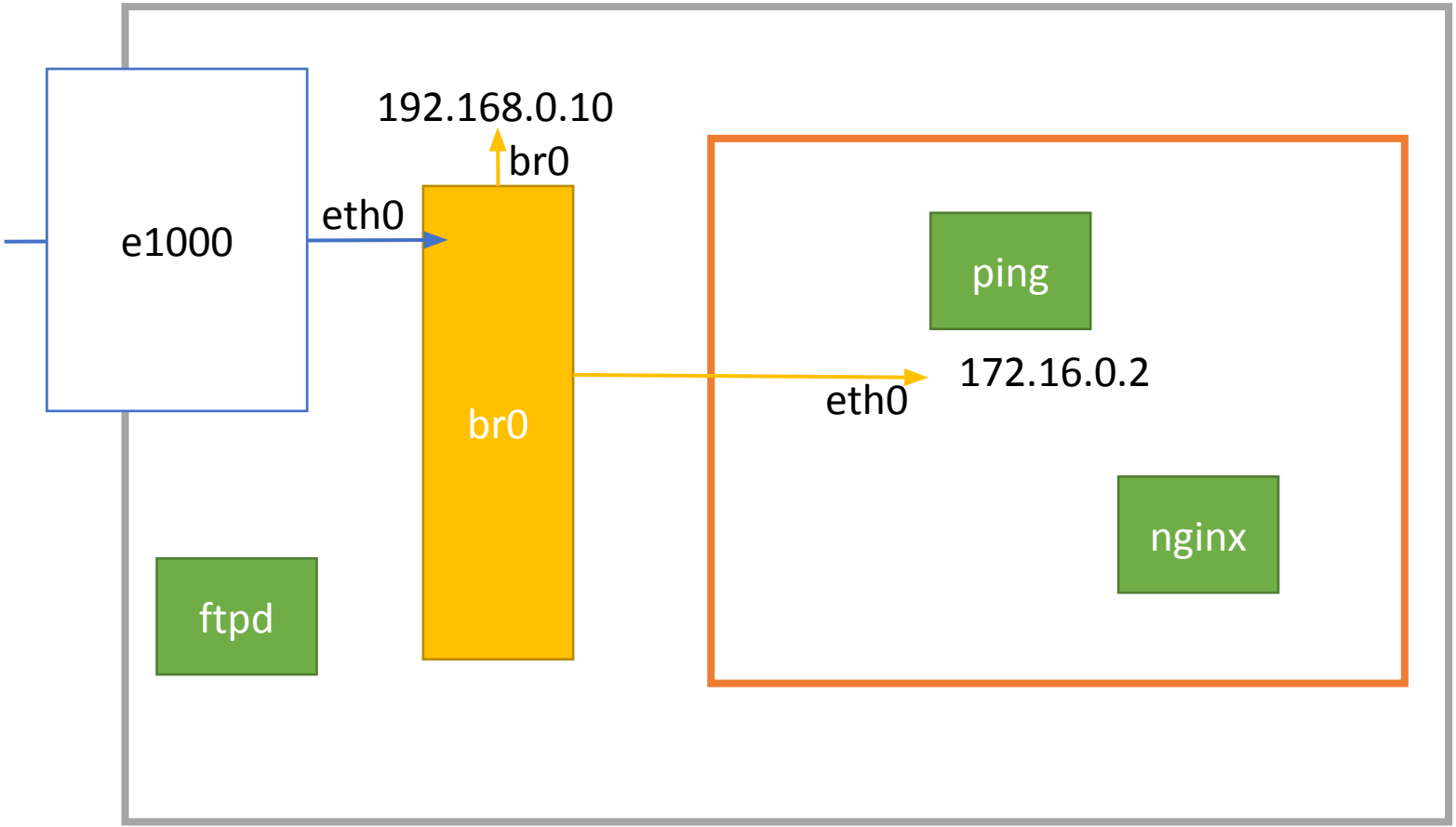

e1000
192.168.0.10
br0
eth0
br0
ping
nginx
veth0
veth1

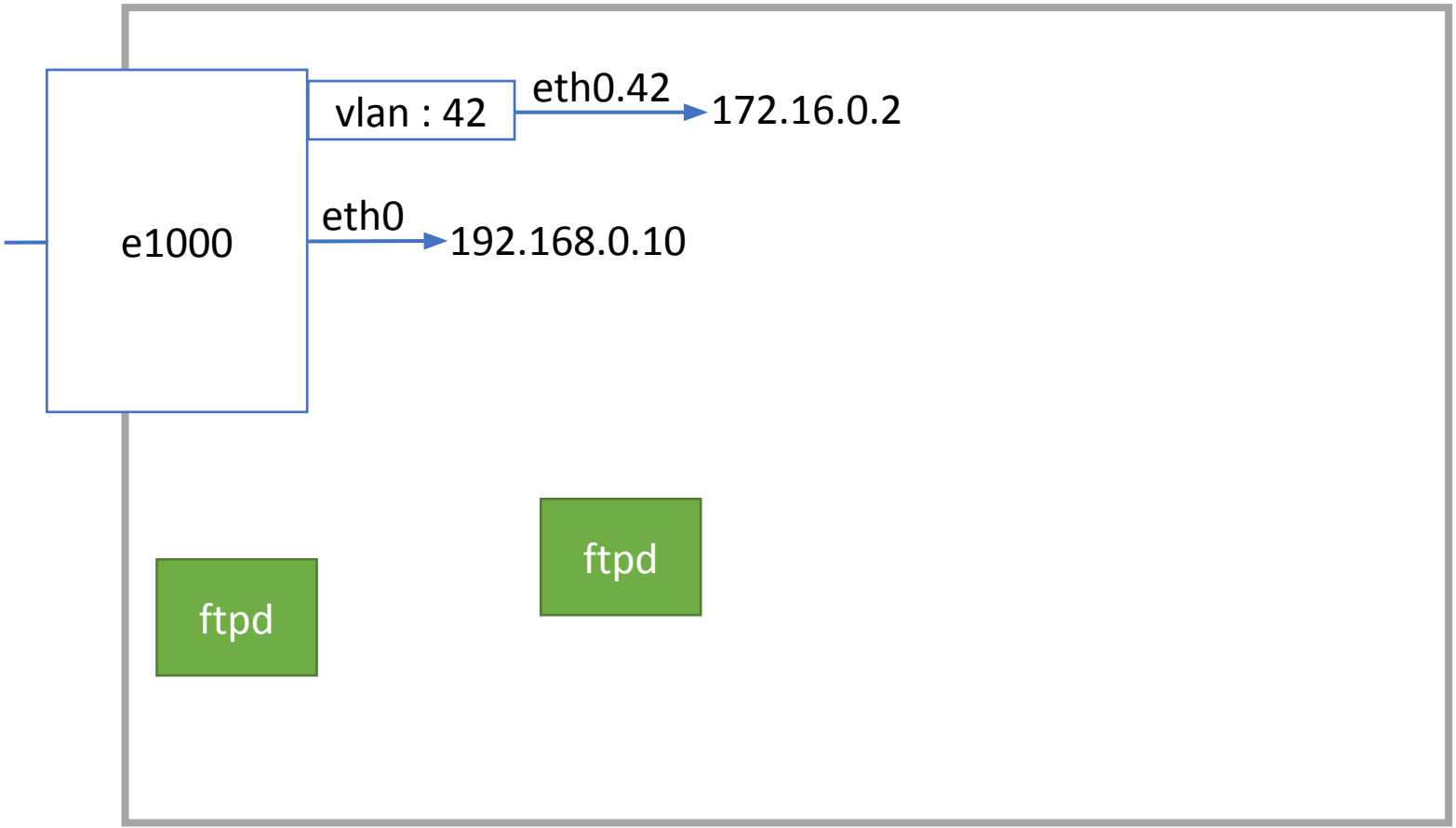

e1000
192.168.0.10
br0
eth0
br0
ping
nginx
172.16.0.1
veth0
veth1
172.16.0.2

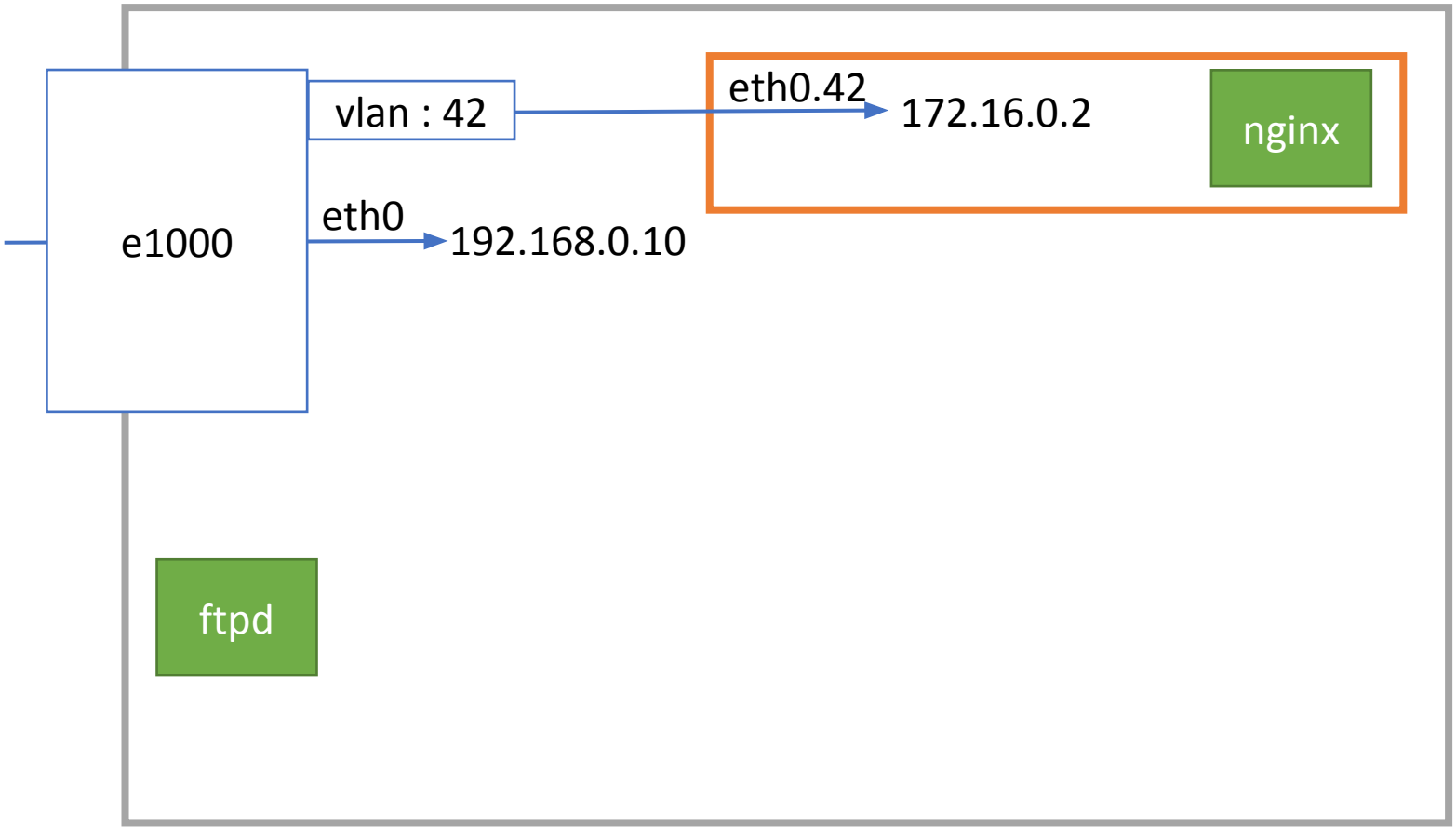

e1000
192.168.0.10
br0
eth0
br0
veth0
ping
nginx
veth1
172.16.0.2

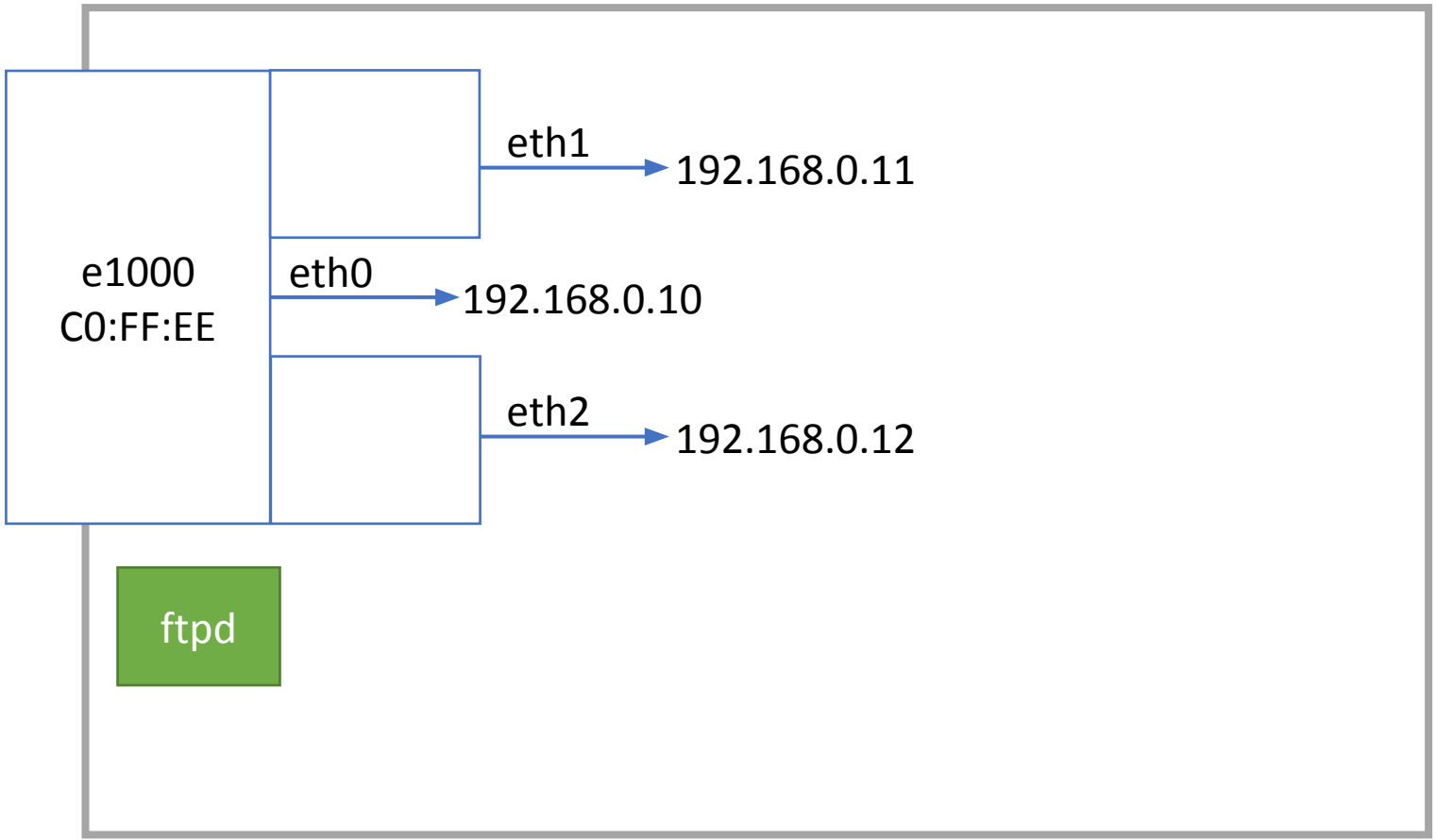

e1000
192.168.0.10
br0
eth0
br0
ping
veth0
172.16.0.2
veth1
nginx
ftpd

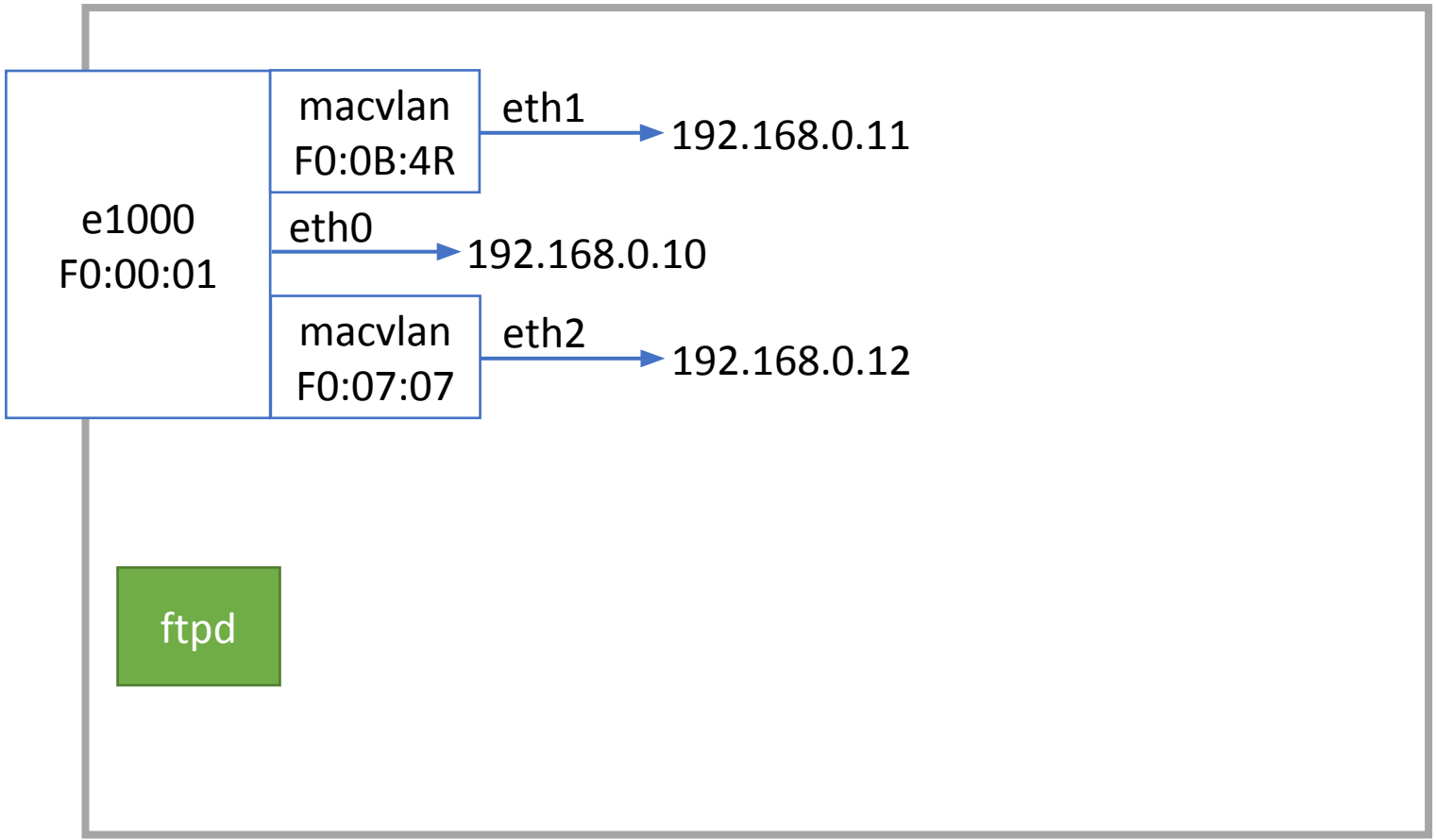

e1000
192.168.0.10
br0
eth0
br0
ftpd
ping
veth0
172.16.0.2
eth0
nginx

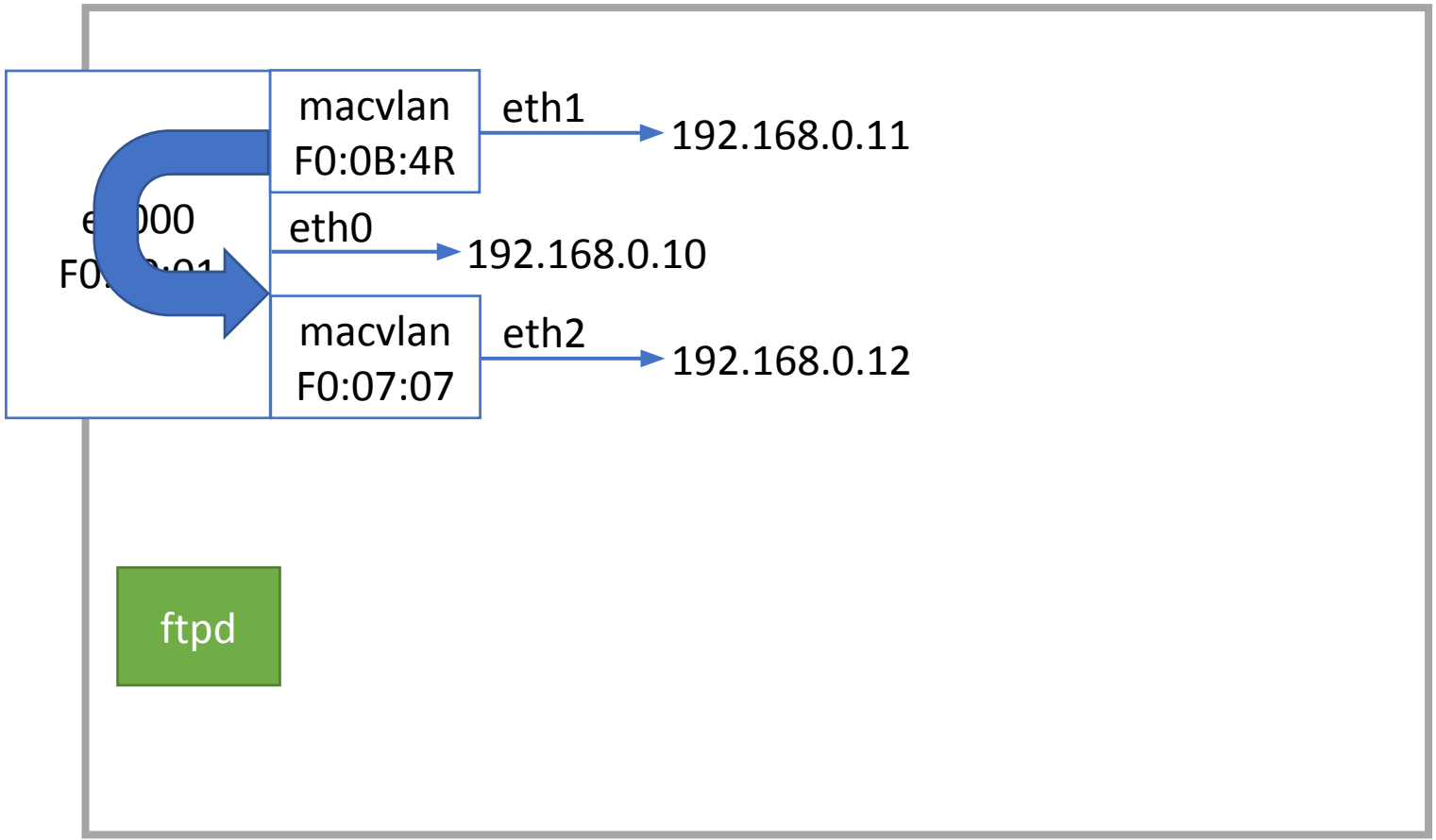

e1000
192.168.0.10
br0
eth0
br0
ping
172.16.0.2
eth0
nginx
ftpd

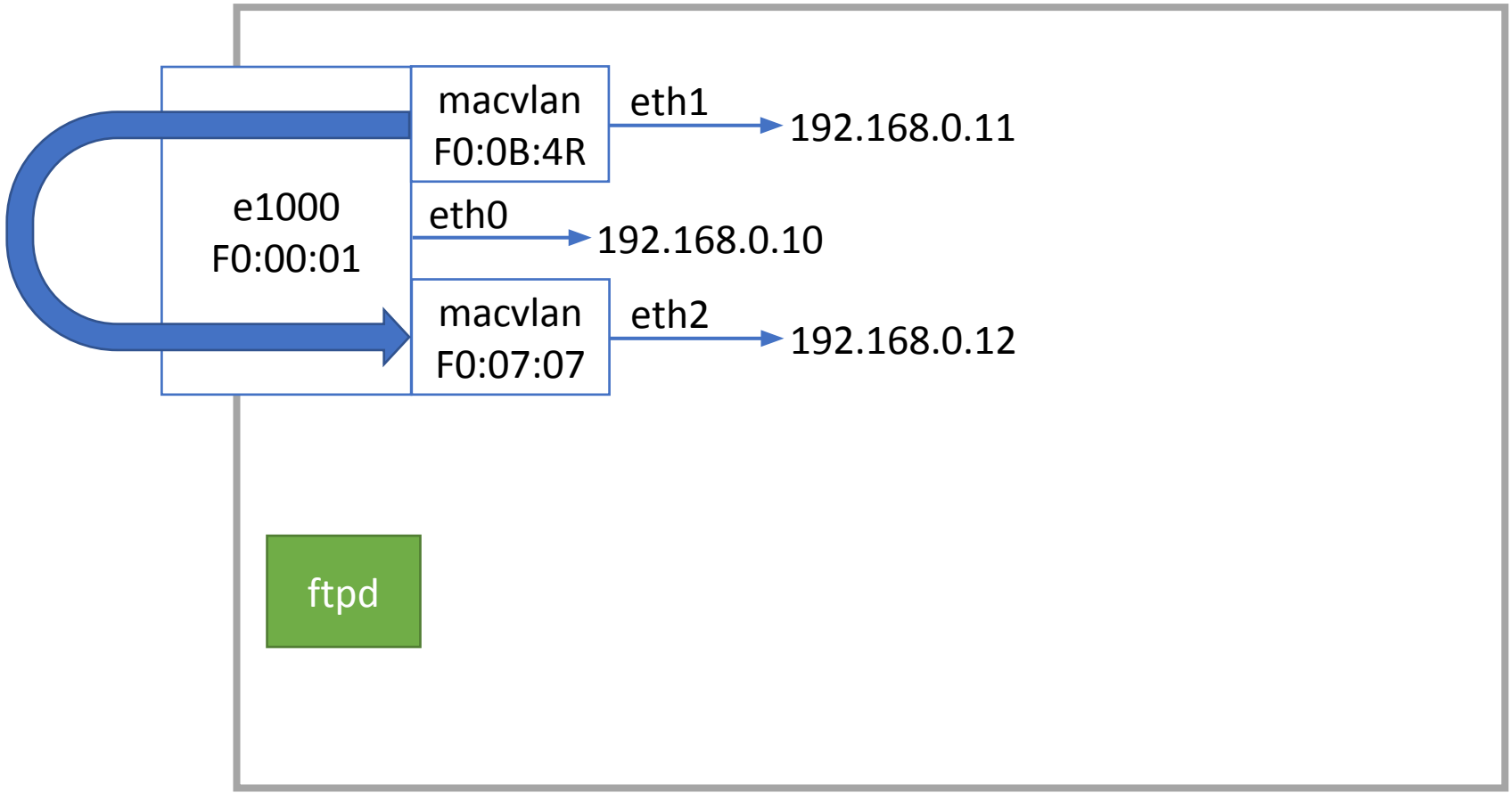

e1000
vlan : 42
eth0.42
172.16.0.2
eth0
192.168.0.10
ftpd
ftpd

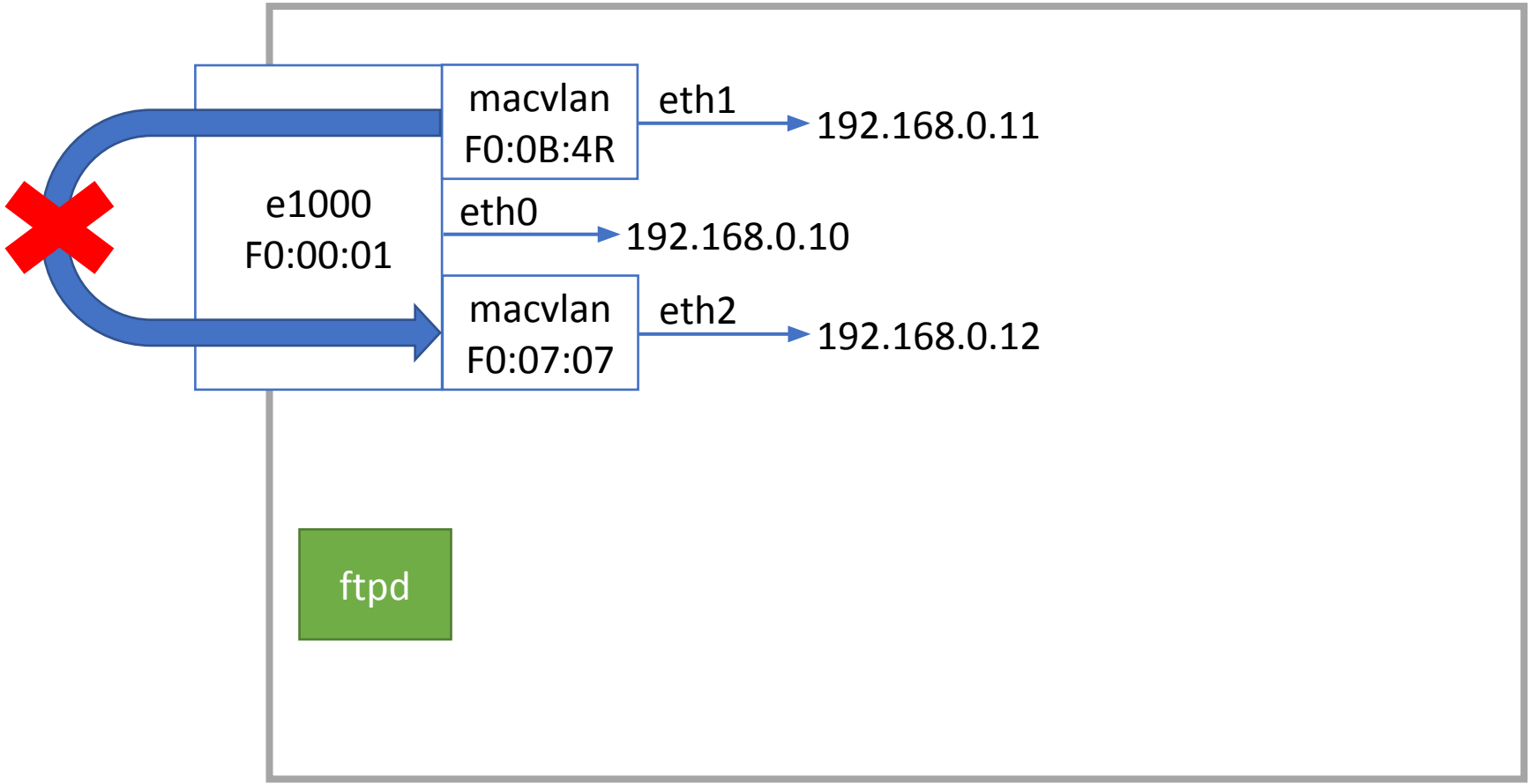

e1000
vlan : 42
eth0.42
172.16.0.2
nginx
eth0
192.168.0.10
ftpd

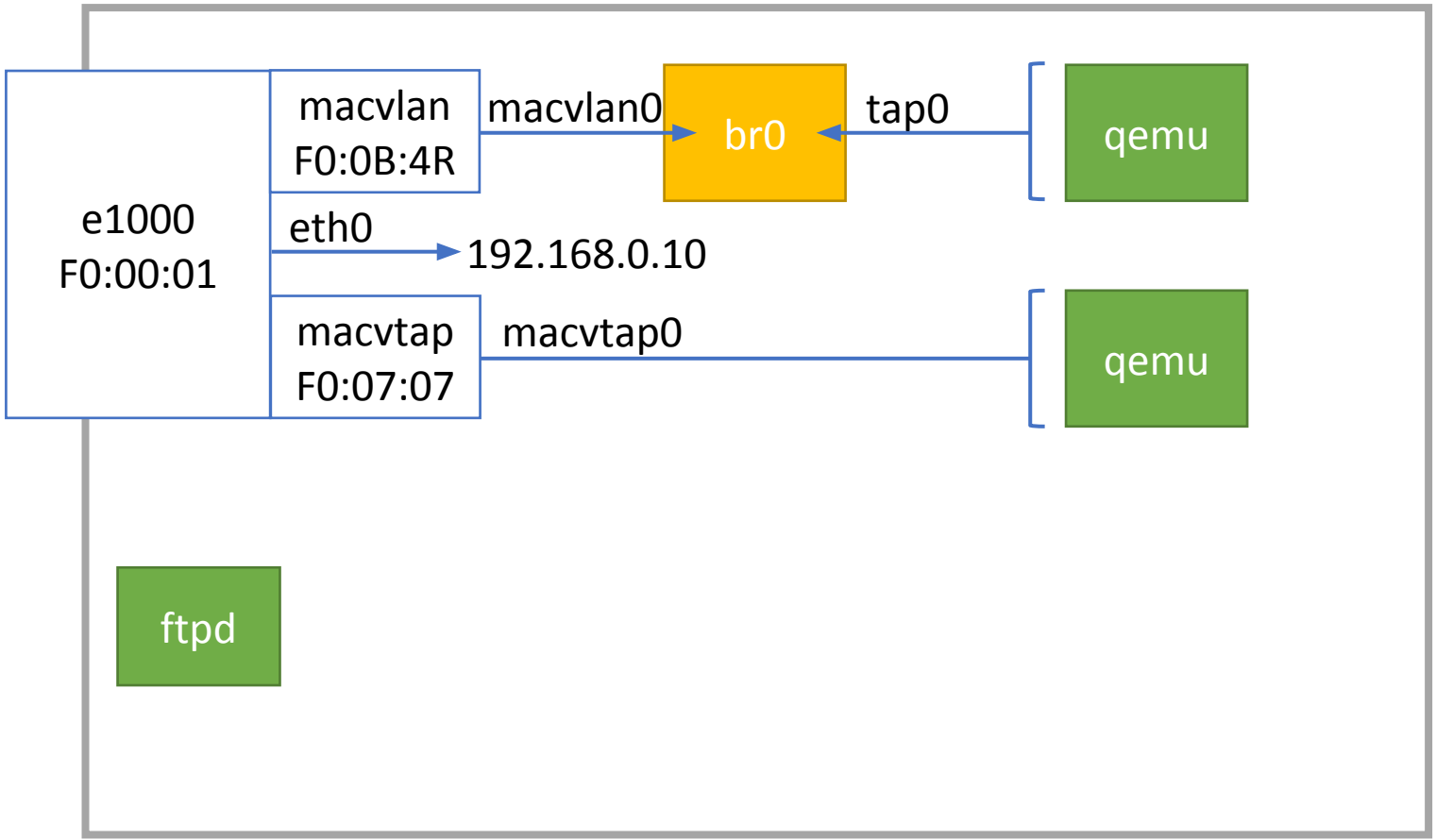

e1000
C0:FF:EE
eth1
192.168.0.11
eth0
192.168.0.10
eth2
192.168.0.12
ftpd

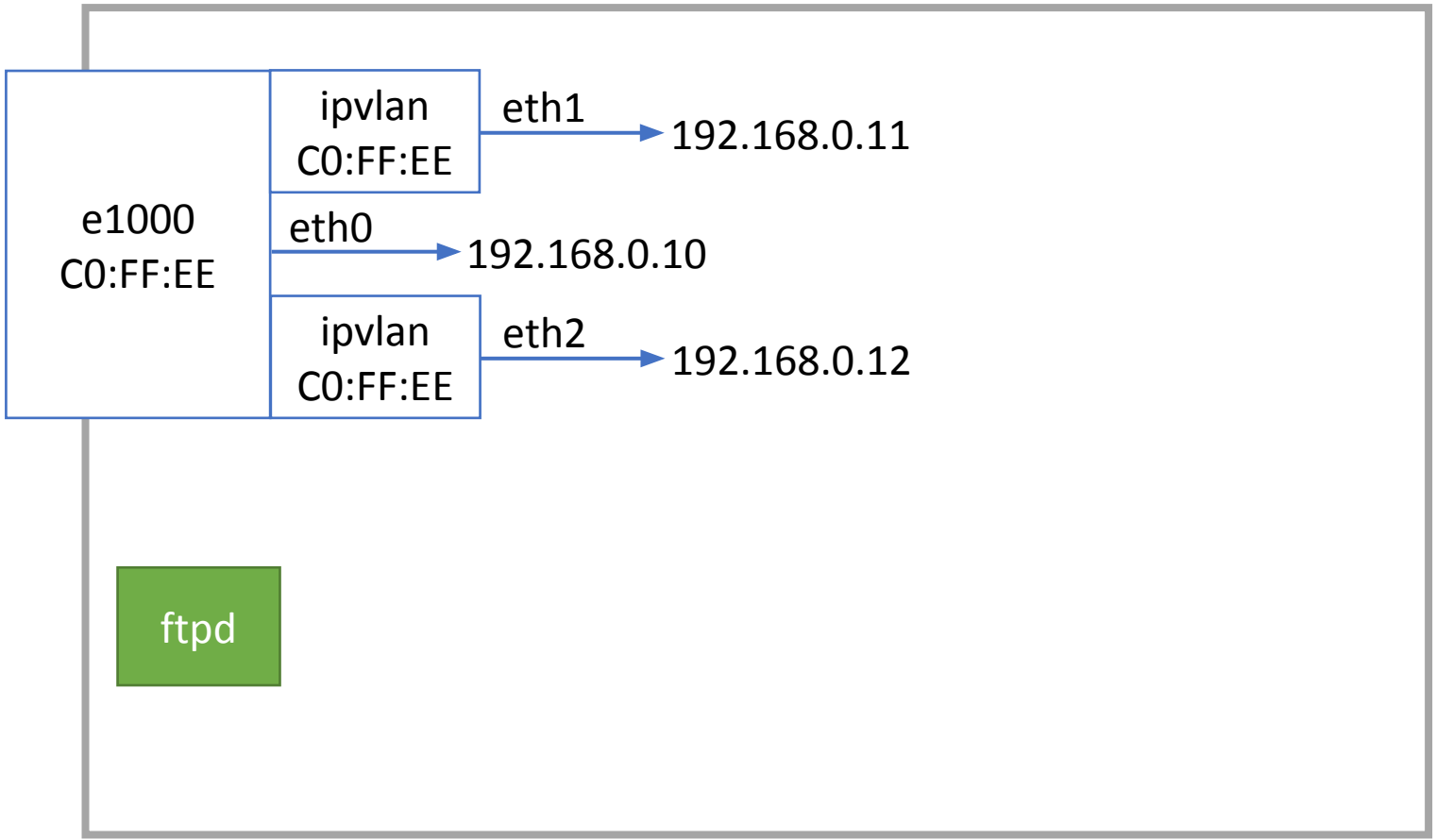

e1000
F0:00:01
macvlan
F0:0B:4R
eth1
192.168.0.11
eth0
192.168.0.10
macvlan
F0:07:07
eth2
192.168.0.12
ftpd

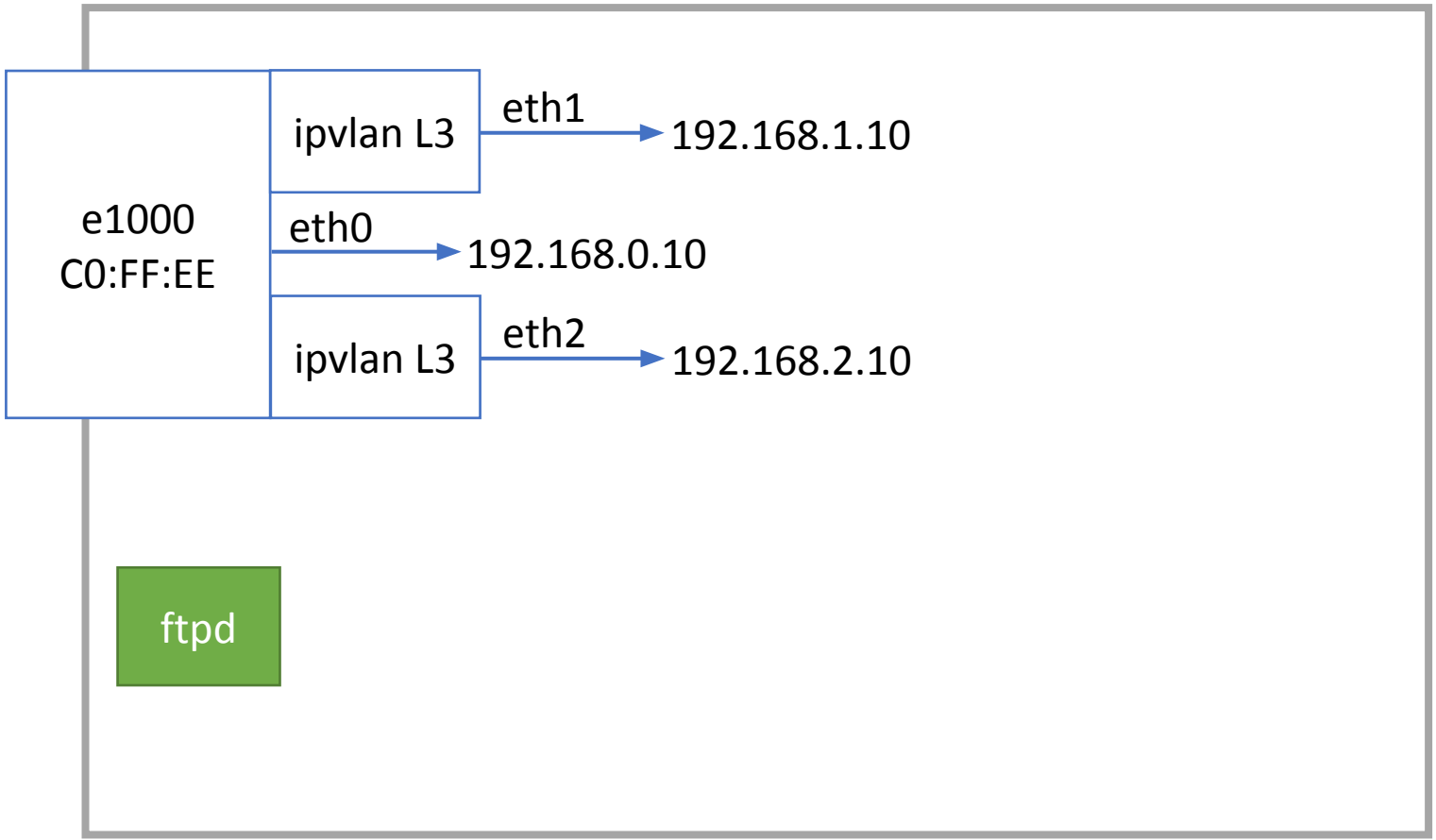

macvlan
F0:0B:4R
eth1
192.168.0.11
eth0
192.168.0.10
macvlan
F0:07:07
eth2
192.168.0.12
ftpd

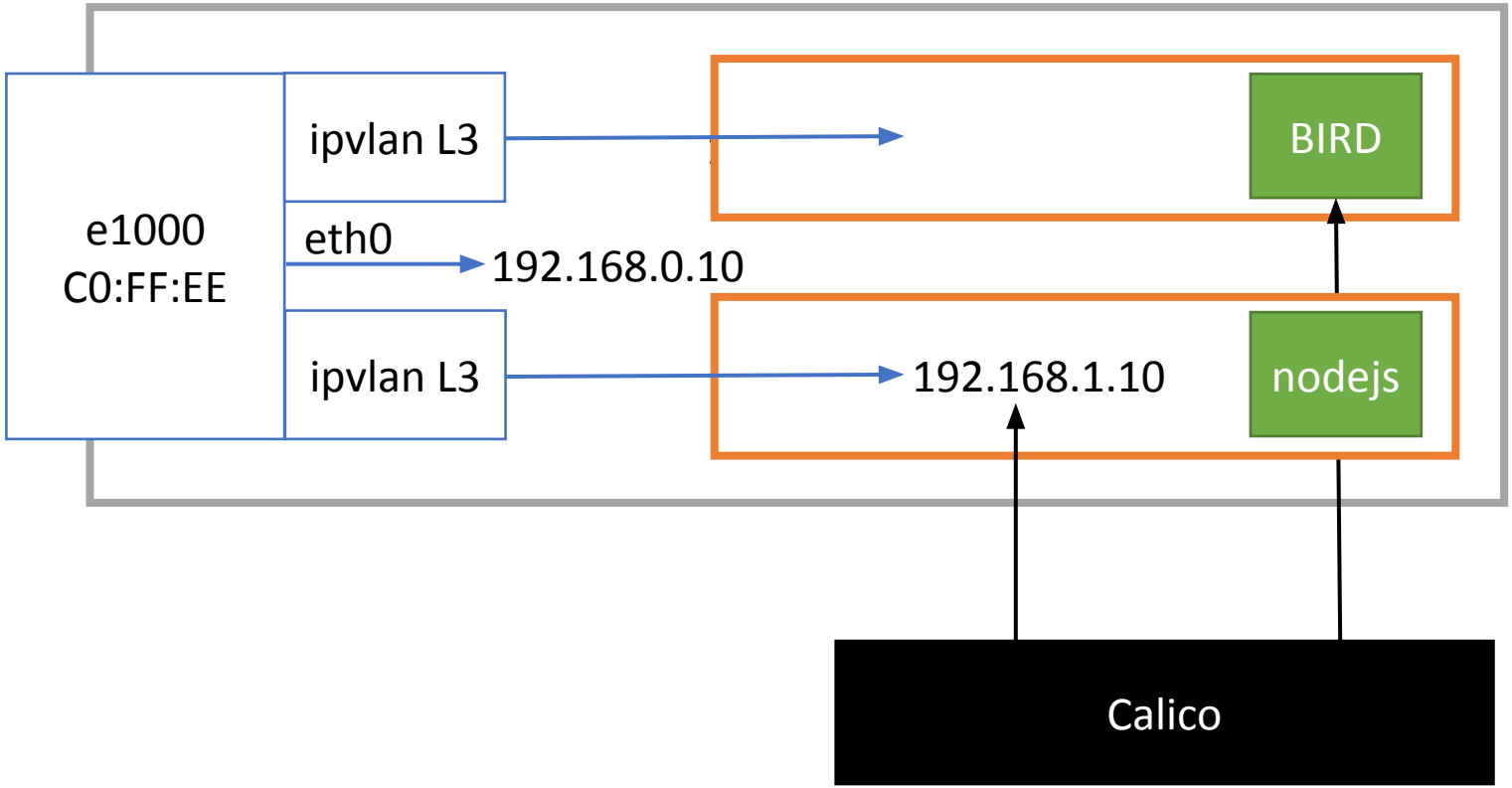

macvlan
F0:0B:4R
eth1
192.168.0.11
e1000
F0:00:01
eth0
192.168.0.10
macvlan
F0:07:07
eth2
192.168.0.12
ftpd

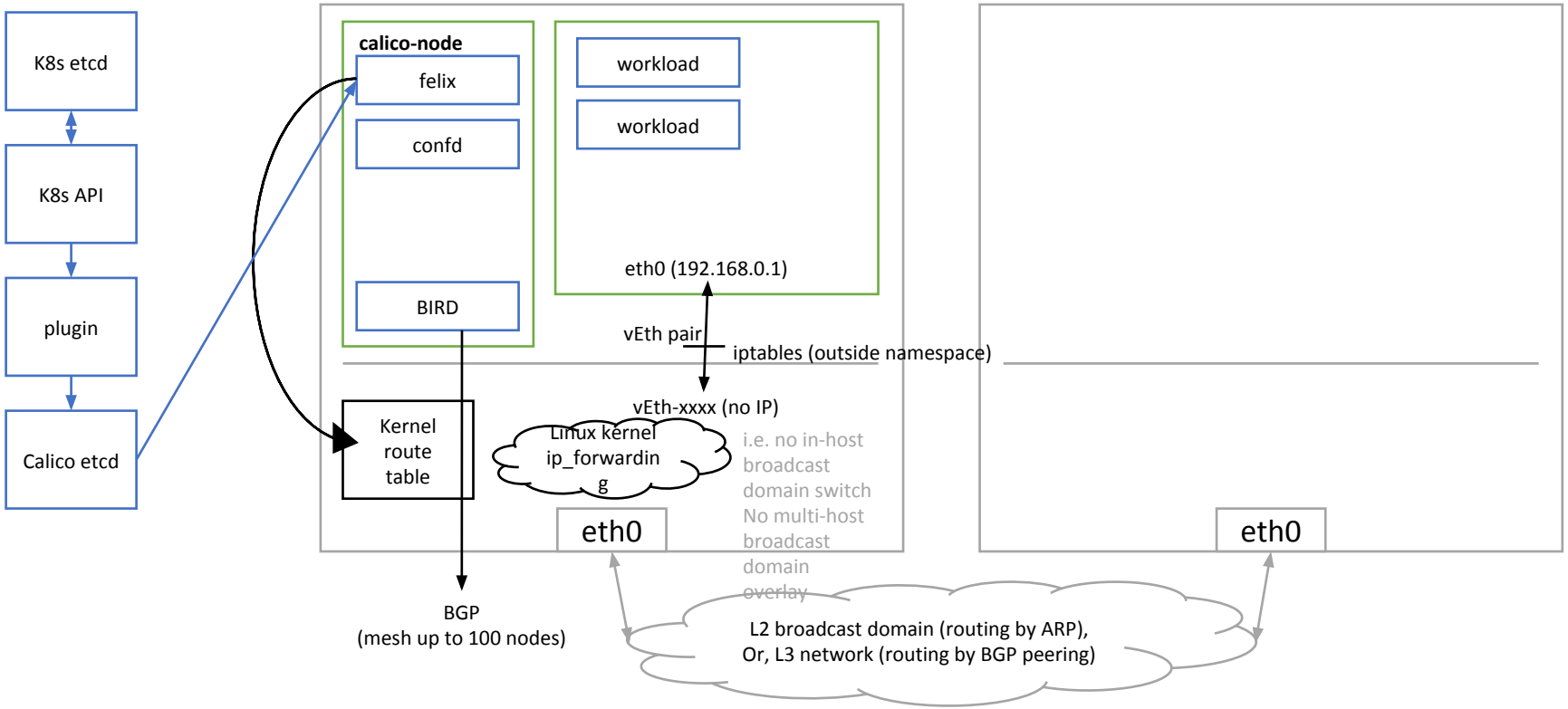

macvlan
F0:0B:4R
eth1
192.168.0.11
e1000
F0:00:01
eth0
192.168.0.10
macvlan
F0:07:07
eth2
192.168.0.12
ftpd


e1000
F0:00:01
macvlan
F0:0B:4R
macvlan0
br0
tap0
qemu
eth0
192.168.0.10
macvtap
F0:07:07
macvtap0
qemu
ftpd


e1000
C0:FF:EE
ipvlan
C0:FF:EE
eth1
192.168.0.11
eth0
192.168.0.10
ipvlan
C0:FF:EE
eth2
192.168.0.12
ftpd


e1000
C0:FF:EE
ipvlan L3
eth1
192.168.1.10
eth0
192.168.0.10
ipvlan L3
eth2
192.168.2.10
ftpd


e1000
C0:FF:EE
ipvlan L3
eth0
192.168.0.10
ipvlan L3
192.168.1.10
BIRD
nodejs
Calico


K8s etcd
K8s API
plugin
Calico etcd
calico-node
felix
confd
BIRD
workload
workload
eth0 (192.168.0.1)
vEth pair
iptables (outside namespace)
vEth-xxxx (no IP)
Kernel route table
Linux kernel ip_forwarding
i.e. no in-host broadcast domain switch
No multi-host broadcast domain
overlay
eth0
eth0
BGP
(mesh up to 100 nodes)
L2 broadcast domain (routing by ARP),
Or, L3 network (routing by BGP peering)

Recap - Interface Types

- Loopback
- Dummy
- “real”
- (multiple L3 addresses)
- TAP / TUN
- vEth
- Vlan
- Macvlan / macvtap
- Ipvlan / ipvtap L2
- Ipvlan / ipvtap L3

Recap - Bridge Types

- Linux Bridge
- Open vSwitch (inc DPDK)
- Macvtap
- (netmap / VALE)
- (snabbswitch)
- (SR-IOV NIC, Cisco vNIC)

Thanks!

@mt165

Slides
Videos
Demo code

mt165.co.uk